

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau



INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

| | | |
|--|-----------|--|
| (51) International Patent Classification ⁵ : G06F 13/00 | A1 | (11) International Publication Number: WO 93/19422 (43) International Publication Date: 30 September 1993 (30.09.93) |
|--|-----------|--|

(21) International Application Number: PCT/US93/02839
(22) International Filing Date: 25 March 1993 (25.03.93)
(30) Priority data:
07/857,578 25 March 1992 (25.03.92) US

(71) Applicant: ENCORE COMPUTER U.S., INC. [US/US];
6901 West Sunrise Boulevard, Fort Lauderdale, FL
33313 (US).

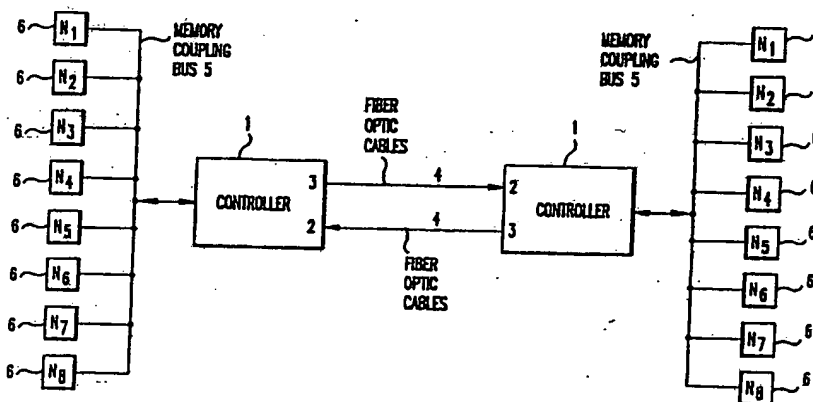
(72) Inventors: KENT, Steven ; 1171 NW 133rd Terrace, Sun-
rise, FL 33323 (US). SCHELONG, Steven ; 780 W. Plan-
tation Circle, Plantation, FL 33324 (US).

(74) Agent: FLEIT, Martin; Keck, Mahin & Cate, P.O. Box
06110, Chicago, IL 60606-0110 (US).

(81) Designated States: AT, AU, BB, BG, BR, CA, CH, CZ,
DE, DK, ES, FI, GB, HU, JP, KP, KR, LK, LU, MG,
MN, MW, NL, NO, PL, RO, RU, SD, SE, SK, Euro-
pean patent (AT, BE, CH, DE, DK, ES, FR, GB, GR,
IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF,
CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

Published
With international search report.

(54) Title: FIBER OPTIC MEMORY COUPLING SYSTEM



(57) Abstract

A system for coupling sets of plurality of nodes (6) that are memory coupled to pass write only data memory to memory via a data link (4) that further includes an optical fiber controller (1) coupled to each data link (4). Each controller (1) is interconnected through fiber (4) for high speed data transfers from one set of nodes (6) to another. The controller (1) is capable of connection implementing three and four cable interfaces. The data is transmitted through the fiber (4) serially but the controller (1) is adapted to receive parallel data and convert to serial form and vice versa.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

| | | | | | |
|----|--------------------------|----|---------------------------------------|----|--------------------------|
| AT | Austria | FR | France | MR | Mauritania |
| AU | Australia | GA | Gabon | MW | Malawi |
| BB | Barbados | GB | United Kingdom | NL | Netherlands |
| BE | Belgium | GN | Guinea | NO | Norway |
| BF | Burkina Faso | GR | Greece | NZ | New Zealand |
| BG | Bulgaria | HU | Hungary | PL | Poland |
| BJ | Benin | IE | Ireland | PT | Portugal |
| BR | Brazil | IT | Italy | RO | Romania |
| CA | Canada | JP | Japan | RU | Russian Federation |
| CF | Central African Republic | KP | Democratic People's Republic of Korea | SD | Sudan |
| CG | Congo | KR | Republic of Korea | SE | Sweden |
| CH | Switzerland | KZ | Kazakhstan | SK | Slovak Republic |
| CI | Côte d'Ivoire | LJ | Liechtenstein | SN | Senegal |
| CM | Cameroun | LK | Sri Lanka | SU | Soviet Union |
| CS | Czechoslovakia | LU | Luxembourg | TD | Chad |
| CZ | Czech Republic | MC | Monaco | TG | Togo |
| DE | Germany | MG | Madagascar | UA | Ukraine |
| DK | Denmark | MI | Mali | US | United States of America |
| ES | Spain | MN | Mongolia | VN | Viet Nam |
| FI | Finland | | | | |

FIBER OPTIC MEMORY COUPLING SYSTEM

FIELD OF THE INVENTION

This invention relates to a novel fiber optic memory interconnection for linking special memory busses of processing nodes and has particular application for real time data processing systems operating at large distances.

BACKGROUND OF THE INVENTION

Systems for updating memories in coupled nodes are known from U.S. Patent No. 4,991,079 the content of which is here incorporated by reference and from U.S. Serial No. 07/403,779 filed September 8, 1989, which is a continuation of Serial No. 06/880,222 filed June 30, 1986, now abandoned, the content of which is here incorporated by reference, all of which are commonly owned with the present application. Such systems use two ported memories and are used to transfer writes to one memory in one node automatically and at high speed to memory in other nodes with the intervention of a CPU. Such systems, however, have a distance limitation of about 120 feet and eight nodes. The present invention is an improvement that enables such systems to be connected over a distance. The present state of the art allows for connections of 3 kilometers and up to ten kilometers with a high speed data interface.

SUMMARY OF THE INVENTION

The present invention provides a means for connecting such memory coupled processing systems over a large distance and provides for high speed data transfers between the systems, copying data from the memory of a node in one system to the memory of a node in another system.

Other and further advantages of the present invention will become readily evident from the following description of a preferred embodiment when taken in conjunction with the appended drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 illustrates the system of the present invention;

Figure 2 illustrates the fiber to memory coupled system controller (FMC);

Figure 3 illustrates the FMC to memory coupling bus interface;

Figure 4 illustrates the method of handling a memory write transfer by the FMC;

Figure 5 illustrates an example of address translation as performed by the present invention;

Figure 6 illustrates the FMC separated into data path quadrants;

Figure 7 illustrates Quadrant 0 of the FMC;

Figure 8 illustrates Quadrant 1 of the FMC;

Figure 9 illustrates Quadrant 2 of the FMC;

Figure 10 illustrates Quadrant 3 of the FMC;

Figure 11 illustrates a parallel/serial latch used in the present invention;

Figure 12 illustrates the basic packet format;

Figure 13 illustrates the format for a data packet;

Figure 14 illustrates the format for a general purpose async data packet;

Figure 15 illustrates the format for a data packet of a MCS-II multidrop console;

Figure 16 illustrates the format for an interrupt packet;

Figure 17 illustrates the mode of operation of the FMC for the handling of async serial data;

Figure 18 illustrates the mode of operation for handling special async data pass through;

Figure 19 illustrates handling of interrupts by the FMC;

Figure 20 illustrates an example of network linking clusters of nodes;

Figure 21 illustrates a secondary backup high speed link for the system;

Figure 22 illustrates internal loopback for memory write transfers;

Figure 23 illustrates external loopback for memory write transfers;

Figure 24 illustrates memory coupling bus loopback;

Figure 25a illustrates internal loopback for general purpose async data;

Figure 25b illustrates external loopback for general purpose async data;

Figure 25c illustrates port-port loopback;

Figure 26 illustrates cluster FMC error handling;

Figure 27 illustrates hub FMC error handling;

Figure 28 illustrates typical configurations of the present invention;

Figure 29 illustrates a star configuration of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

This invention relates to a high speed data interface system using fiber optic memory interconnection as shown in Figure 1. This interface system connects memory coupled systems over distances up to 10 kilometers. Each memory coupled system comprises a data link or memory coupling bus 5 to which up to eight nodes 6 are coupled each comprising a processor, I/O, a two or more ported memory and a processor to memory bus. Write/read sense controllers couple the busses and the memory so writes only are sensed and reflected to the memories of other nodes without CPU intervention. This is described in detail in the patent and application noted above, both of which are here incorporated by reference. The bus 5 of each memory coupled system is connected to a fiber-to-memory coupling system controller (FMC) 1. Each FMC has both an input and output port for connection with another FMC. The input port 2 is for receiving transmitted data from another memory coupled system and the output port 3 is for transmitting data to another memory coupled system. The transmission of the data is through fiber optic cables 4.

The Fiber Optic Memory Coupling System (FOMCS) also provides

1
2 the capability for nodes on separate MC busses to exchange async
3 serial data across the fiber link and for a node on one MC bus to
4 interrupt a node on another. Each FMC supports 8 general purpose
5 async ports and 16 interrupt lines (8 input and 8 output). The
6 multidrop console link incorporated in the MCS-II fourth cable is
7 also supported. These features are intended to allow remote
8 booting of nodes across the fiber and to provide a means of
9 synchronizing the actions of nodes.

10 Throughout this document, the term "MCS cluster" (or simply
11 "cluster") is used to refer to a MC bus and its attached nodes and
12 FMC. In configurations of only two or three clusters, pairs of
13 FMCs are used to directly connect MC busses (see Figure 28). In a
14 configuration of more than three clusters, a FOMCS hub is used to
15 connect all the clusters in a star configuration (see Figure 29).

16 The configuration programming link shown in Figures 28 and 29
17 is used to establish the operational mode of each FMC. FMCs which
18 are not directly connected to the programming link receive
19 programming information via packets sent over the fiber link. The
20 FMC 1 can be seen in more detail in Figure 2. The FMC includes a
21 Receive Data Path 7, an output latch 9, a receive FIFO 11, a
22 receive Error Detection Circuit 13, two receive latches 15 and 17,
23 a receiver 19, an input latch 10, a hit and translation RAM 12, a
24 transmit FIFO 14, a transmit Error Detection Circuit 16, two
25 transmit latches 18 and 20 and a transmitter 22.

26 The FMC 1 consists of four main sections: the memory coupled
27 system 5 interface, the data paths (Rx and Tx) 7 and 8, the high
28 speed serial data link interface 4, and the microprocessor. These
29 areas are delineated using dashed lines in the functional block
30 diagram (Figure 2). The Fiber Transition Module (FTM) connects to
31 the FMC high speed serial link interface. The FTM is a known and
32 conventional piece of hardware and serves to connect the electrical
33 signals to corresponding light signals and vice versa.

34 The Memory Coupling (MC) bus interface is the FMC's link to
35 the Memory Coupling System (MCS) and the nodes on that bus. The
36 FMC implements a three-cable, 90-signal interface for those MCS

1
2 networks that use a standard MC bus (24-bit addresses), and a
3 four-cable, 120-signal interface for those MCS networks that use
4 the 28-bit addresses provided in MCS-II. Provision is also made in
5 the four-cable interface for eventual support of 32-bit addresses.
6 The FMC supports all the address lines defined in the four-cable
7 interface; however, in an environment where addresses are a full 32
8 bits, the FMC will only reflect into and out of the first 256
9 megabytes of memory.

10 The FMC appears as a typical MCS node on the bus, using one
11 of the nine bus IDs (0 - 8) available in MCS or MCS-II. The MCS
12 used by the FMC is set during initialization by the configuration
13 programming link.

14 Details of the FMC interface to the MC bus are illustrated in
15 Figure 3. The FMC MC bus interface receives bus transfers by
16 latching the memory address, memory data, flag bits and parity on
17 the rising edge of DATA VALID (assuming DATA VALID is not being
18 driven by the FMC itself). The two different bit counts for the
19 address, flags and parity lines reflect differences between the
20 three cable MCS bus and the four cable MCS-II bus. The smaller
21 counts apply to the three cable bus. Note that 32 bits of address
22 are indicated for the four cable MCS-II bus rather than 28. As
23 mentioned earlier, signals to support the additional four address
24 bits are reserved in the fourth cable for future expansion. One
25 bit of odd parity is provided for each byte of address and data,
26 resulting in the seven or eight parity bits shown in Figure 3.

27 The flag bits qualify the received transfer. For memory
28 write transfers, one flag bit (referred to as the "F-bit")
29 indicates whether the memory write is to a byte location in memory
30 as opposed to a halfword or word. The other two flag bits, which
31 are only present in the MCS-II bus, differentiate between memory
32 write transfers and other types of transfers. While only memory
33 write transfers appear on an MCS-II bus in a cluster, the MCS-II
34 bus in the FOMCS hub is used to distribute interrupt, async and
35 other types of data in addition to memory write traffic.

36 As indicated in Figure 3, the address, flags and parity are

1
2 treated as 32, 3 and 8 bit quantities, respectively, when received
3 by the FMC. If a three cable MCS bus is connected to the FMC, the
4 receivers for the eight most significant address bits, the two
5 extra flag bits and the eighth parity bit are disabled. Zeros are
6 put in the MC input latch for the missing address and flag bits and
7 a one is inserted for the missing parity bit. In a four cable
8 environment, 32 bit addresses are received but the FMC will discard
9 any received memory write transfer in which the most significant
10 four address bits are not zero.

11 If receipt of a transfer causes the FMC Tx FIFO 14 to become
12 half full, the FMC will drive GLOBAL BUSY on the MC bus to prevent
13 overflow. GLOBAL BUSY is deasserted when the Tx FIFO 14 becomes
14 less than half full. The CONTROL BUSY signal is only present in
15 the MCS-II bus and is only utilized in the FOMCS hub. If receipt
16 of a non-memory-write type transfer causes the FMC FIFO used to
17 hold such transfers to become half full, the FMC will drive CONTROL
18 BUSY to prevent overflow. When the FIFO becomes less than half
19 full, CONTROL BUSY is deasserted.

20 Before the FMC can generate a transfer on the MC bus, it must
21 first acquire the right to access the bus. To accomplish this, the
22 FMC asserts the REQUEST line on the bus which corresponds to the
23 MCS ID the FMC has been programmed to use. The FMC then monitors
24 the corresponding GRANT line. When the bus arbiter asserts the
25 GRANT line, the FMC drives the memory address, memory data, flag
26 bits and parity on the bus. The FMC's MCS ID is also driven on the
27 node id lines of the bus. On the next rising edge of the VALID
28 ENABLE signal, the FMC drives DATA VALID. (The VALID ENABLE signal
29 is a free running clock generated by the bus arbiter to synchronize
30 the actions of nodes on the bus.)

31 Note that the MC output latch illustrated in Figure 3 only
32 supplies 28 bits of the memory address rather than 32. The driver
33 inputs for the remaining four address bits are tied to ground
34 (i.e., the bits are forced to be zeros).

35 If receipt of a memory write transfer packet causes the FMC
36 Rx FIFO 11 to become half full and MC bus burst request mode is

1
2 enabled, the FMC will drive BURST REQUEST. Asserting the BURST
3 REQUEST signal, which is only present in the MCS-II bus, causes the
4 bus arbiter to enter a mode in which only the FMC is granted access
5 to the bus. After asserting BURST REQUEST and delaying long
6 enough to guarantee propagation of the signal to the arbiter, the
7 FMC deasserts GLOBAL BUSY. If no other node on the bus is
8 asserting GLOBAL BUSY, the arbiter will begin issuing grants to the
9 FMC. If one or more other nodes is asserting GLOBAL BUSY, the
10 arbiter waits for the busy condition to clear and then begins
11 issuing grants to the FMC. Note that the FMC can safely deassert
12 GLOBAL BUSY even if it is unable to accept more transfers from the
13 bus because the arbiter will only grant bus access to the FMC.

14 When the FMC has unloaded enough packets front its Rx FIFO 11
15 to cause the fill level to drop below half full, BURST REQUEST is
16 deasserted and the arbiter returns to the normal "fairness"
17 arbitration scheme. If the FMC is unable to accept more transfers
18 from the bus, it will assert GLOBAL BUSY before deasserting BURST
19 REQUEST.

20 If the FMC's Tx FIFO is less than half full, the FMC will
21 keep BURST REQUEST asserted for 8 MC bus cycles and then release it
22 for 16 cycles and then assert it for 8, release it for 16 and so on
23 until the Rx FIFO fill level drops below half full. If both the
24 Rx and Tx FIFOs are half or more full, the FMC will assert BURST
25 REQUEST continuously until enough packets have been unloaded from
26 the Rx FIFO to cause the fill level to drop below half full. Once
27 BURST REQUEST is deasserted, the arbiter returns to the normal
28 "fairness" arbitration scheme. If the FMC is unable to accept more
29 transfers from the bus, it will assert GLOBAL BUSY before
30 deasserting BURST REQUEST.

31 Support for burst request mode in the FMC is enabled or
32 disabled via a configuration programming command. Only one FMC on
33 a given MC bus can have burst request mode enabled but at least one
34 FMC in every FMC-to-FMC link must have the mode enabled to ensure
35 reliable operation. Note that enabling burst request mode in the
36 FMC only means that the FMC is able to drive BURST REQUEST on the

bus. For burst request mode to be useful, the four cable MCS-II environment with Memory Coupling Controllers (MCC's) as the bus arbiter/terminators is required. MCC's provide bus termination and arbitration in a conventional manner as known from the previously noted patent and application and are sometimes referred to as reflective memory controllers.

The FMC Rx and Tx data paths 7 and 8 move data between the MCS interface and the high speed serial data link. They also interface to the microprocessor allowing asynchronous data, interrupt transfers and flow control information to move between clusters.

Figure 4 illustrates the manner in which the FMC processes MCS memory write transfers. As indicated, not all transfers are transmitted as packets over the high speed serial data link. Only those transfers which correspond to memory writes into selected regions of memory are transmitted over the high speed link. Transfers which do not fall within these selected regions are simply discarded.

In the other direction, packets containing memory write transfers are received over the high speed link and decoded by the FMC. Note that in the Rx path 7 there is no hit/translation RAM so all memory write packets received over the high speed link cause memory write transfers to be generated on the MC bus. When a packet has been received which represents a write into memory, the FMC requests the use of the MC bus and when the request is granted, generates a memory write transfer on the bus.

In the Tx path 8, the regions of memory to be reflected are defined during configuration programming of the FMC. During programming, the total memory address space is viewed as a sequence of 8K byte (2K word) blocks. Multiple regions as small as 8K bytes can be defined. Any number of regions, segregated or concatenated, up through the entire address range may be established.

Another feature of the FMC which is illustrated in Figure 4 is memory address translation. Prior to packetizing a memory write transfer and transmitting it over the high speed link, the

memory address is modified. MCS physical addresses are 24 bits long (MCS-I) or 28 bits long (MCS-II). A MCS physical address is translated by using the most significant 11 bits (MCS-I) or 15 bits (MCS-II) to address a hit/translation RAM on the FMC. The value read from the RAM is the most significant 15 bits of the new address.

Note that the least significant 13 bits of a memory address are unaffected by the translation process. Thus, 8k byte blocks are mapped from the address space of the source MC bus to that of the destination MC bus and vice versa. As illustrated in Figure 5, this feature is very useful because it allows clusters to share memory regions which reside in the different places in each cluster's physical address space.

The contents of the hit/translation RAM are established



1 during configuration programming of the FMC. The size of memory
2 addresses (i.e., 24 or 28 bits) on the source MC bus is also
3 established during configuration programming. Each location in the
4 hit/translation RAM represents a 8K byte block of memory and
5 contains a hit bit and the 15 bit translation value. If the hit
6 bit is set, memory writes into the 8K byte memory block are
7 reflected over the high speed link. If the bit is reset, such
8 memory writes are ignored.

9 In the context of the MCS star network configuration, the FMC
10 region selection and address translation features can be used to
11 segregate the MCS clusters connected to the hub into groups such
12 that memory write traffic is only reflected within a group, not
13 between groups. Note that if an administrative system (i.e., not
14 one of the nodes in the star network) is used to program the FMCs,
15 a secure network can be achieved in which cluster groups can
16 coexist but not affect each others memory. If total isolation of
17 groups is not desired, overlapping regions can be used.

18 The FMC microprocessor has the ability to interject data into
19 and remove data from both the Tx and Rx data paths 7 and 8. Such
20 data is referred to as control data to distinguish it from memory
21 write transfers. General purpose async data, MCS-II multidrop
22 console async data and interrupt pulses are all treated as control
23 data.

24 When the FMC receives data over the general purpose async
25 ports, it forms the data into packets and injects those packets
26 into the transmit data stream sent over the high speed link. When
27 a packet of async data is received by the FMC, the bytes are broken
28 out of the packet and sent over the appropriate general purpose
29 async port. MCS-II multidrop console data is handled in a similar
30 fashion.

31 When an enabled input interrupt line is pulsed, the FMC
32 generates a packet that is sent over the high speed link. Upon
33 receipt of an interrupt packet from the serial data link, the FMC
34 pulses the appropriate output interrupt line.

35 Other control transfers include configuration programming

1 information, high availability messages, error indications, and
2 reset indications.

3 Flow control in a MCS network is accomplished by means of
4 flow control bits in the packets sent from FMC to FMC and by means
5 of MC bus busy signals. Internal to the FMC, memory write data
6 transfers and other types of data transfers (i.e., async,
7 interrupt, multidrop, etc.) are handled separately so that flow
8 control may be applied to non-memory write transfers without
9 affecting the memory write traffic. Each packet sent from FMC to
10 FMC contains two flow control bits, one to cause the receiving FMC
11 to cease or resume transmission of memory write transfer packets
12 and another to cause it to cease or resume transmission of other
13 types of packets. This notion of two separate data streams also
14 applies to the hub MC bus where there are separate bus busy
15 signals, one for memory write transfers and another for all other
16 types of transfers.

17 An additional type of flow control is burst request mode.
18 Burst request mode is necessary to ensure that lock ups do not
19 occur on FMC-to-FMC links. When link utilization is high in both
20 directions, the potential exists for a FMC-to-FMC link to lock up
21 in a condition in which both FMCs are asserting busy on their
22 respective MC busses. Because the busy condition on the MC busses
23 prevents the FMCs from generating bus transfers, the FMCs are
24 unable to reduce the fill levels of their Tx FIFOs 14 and will
25 therefore never deassert bus busy.

26 Burst request mode alleviates this problem by allowing a FMC
27 to unload transfers from its Rx FIFO 11 while ensuring that the FMC
28 will not have to accept additional transfers into its Tx FIFO 14.
29 This means the FMC can accept more memory write transfer packets
30 from the remote FMC which in turn allows the remote FMC to unload
31 its Tx FIFO 14 and eventually clear the busy condition on the
32 remote MCS bus. Clearing the busy condition allows the remote FMC
33 to unload its Rx FIFO 11. The remote FMC can then accept more
34 memory write transfer packets which allows the local FMC to unload
35 its Tx FIFO 14 and clear the busy condition on the local MC bus.

1 As shown in Figure 6, the FMC data paths are logically
2 subdivided into quadrants. In quadrant 0, MC bus transfers are
3 received and moved to the Tx FIFO 14. The Tx FIFO 14 serves as the
4 boundary between quadrants 0 and 2. In quadrant 2, transfers
5 removed from the Tx FIFO 14 are packetized and transmitted over the
6 high speed serial link 4.

7 Packets are received over the high speed link 4 in quadrant
8 3 and the packet contents are moved to the Rx FIFO 11. The Rx
9 FIFO 11 serves as the boundary between quadrants 3 and 1. In
10 quadrant 1, information removed from the Rx FIFO 11 is used to
11 generate transfers on the MC bus.

12 An important feature of the FMC design, particularly from a
13 diagnostic perspective, is that the latches in each quadrant can
14 be accessed in both a parallel and serial fashion. During normal
15 operation of the FMC, data moves through the latches using the
16 parallel interface. The alternate serial interface allows the
17 microprocessor on the FMC to shift data serially into and out of
18 the latches.

19 Figure 7 shows a detailed view of quadrant 0. Processing
20 begins in quadrant 0 when the control logic 25 detects that the MC
21 input latch 10 has been loaded. The input latch is unloaded and
22 odd parity is computed on the address and data. The computed
23 parity is then compared to the received parity. At the same time,
24 the most significant address four bits are checked to see if any
25 of them are non-zero.

26 While the parity and address checks are taking place, 15 bits
27 of the address are used to address the hit/translation RAM 12. (If
28 the address came from a three cable MCS bus, the upper four of the
29 15 bits used to address the RAM will be zeros.) The least
30 significant 15 bits of the 16 bit value read from the RAM become
31 the new, or translated, address bits. The most significant bit
32 of the value read from RAM is the window hit bit.

33 The destination of the MC transfer is determined by the
34 parity and address checks, the hit bit, and the flag bits received
35 with the transfer. If any of the most significant four address bits

1 is non-zero, the transfer is discarded. If the received parity
2 does not equal the computed parity, the transfer is clocked into
3 the microprocessor interface FIFO 21. The transfer is also placed
4 into the micro FIFO if the parity is good but the flag bits
5 indicate a control type transfer (which only occurs in the MCS
6 hub). The destination of a memory write type transfer with good
7 parity depends on the setting of the hit bit. If the hit bit is
8 reset, the transfer is simply discarded. If the hit bit is set,
9 the memory data, translated memory address and flag bits are
10 clocked into the Tx FIFO 14.

11 The contents of the hit/translation RAM 12 are initialized
12 by the FMC microprocessor. To change a location in the RAM 12, the
13 microprocessor first puts the new value into the loading buffers
14 24. The microprocessor then causes GLOBAL BUSY to be asserted on
15 the MC bus and waits long enough for any transfer in progress to
16 pass through the MC input latch 10. Then, the microprocessor
17 causes the hit/translation RAM 12 to be written with the value from
18 the loading buffers. GLOBAL BUSY is subsequently deasserted.

19 Figure 8 shows a detailed view of quadrant 1. Quadrant 1
20 processing begins when the control logic 25 detects that the MC
21 output latch 9 is empty and either the Rx FIFO 11 is not empty or
22 the micro interface 21 is requesting use of the latch 9. If there
23 is something in the FIFO 11 and the micro interface 21 is not
24 requesting, the FIFO 11 is read and parity is computed on the
25 memory address and data. The data, address, flags and computed
26 parity are then clocked into the MC output latch.

27 The micro interface 21 is only used when a FMC in a FOMCS hub
28 needs to distribute interrupt, async or other control information
29 to the other FMCs in the hub. If control busy is not asserted on
30 the hub bus, the micro interface logic 21 requests use of the MC
31 output latch 9. When the latch 9 is available, the control data
32 is serially shifted into the latch 9. Odd parity for the transfer
33 is computed by the microprocessor and shifted into the latch 9
34 following the data.

35 Figure 9 shows a detailed view of quadrant 2. Processing

1 begins in quadrant 2 when the control logic detects that the Tx
2 latches 18 and 20 are empty and finds that the Tx FIFO 14 is not
3 empty or that the microprocessor has loaded the micro interface
4 latch. If the micro latch has been loaded, the contents of the
5 latch are transferred to the Tx latches 18 and 20. Of the 72 bits
6 transferred from the micro latch, 64 are also clocked into a pair
7 of EDC (error detection code) generators 16a and 16b. The
8 generated eight bit EDC and the 72 bits from the micro latch are
9 clocked into the Tx latches 18 and 20. The contents of the Tx
10 latches 18 and 20 are then passed to the high speed link serial
11 transmitter 22 in two 40 bit transfers.

12 If the micro latch is empty but the Tx FIFO 14 is not, the
13 memory data, address and flags are read from the FIFO 14. An
14 additional nine bits are generated by the control logic to make a
15 total of 72. Again, an EDC is generated on 64 of the 72 and an 80
16 bit packet is clocked into the Tx latches 18 and 20. The contents
17 of the Tx latches 18 and 20 are then passed to the transmitter 22.

18 If the control logic detects that either the Rx FIFO 11 or the
19 micro FIFO in quadrant 3 are half full, flow control flag bits are
20 asserted in the 80 bit packet clocked into the Tx latches 18 and
21 20. If no packet is available to clock into the latches, the
22 control logic 25 causes a special flow control packet to be clocked
23 into the Tx latches 18 and 20 for transmission to the remote FMC.
24 Separate flow control bits are defined for the Rx FIFO 11 and the
25 quadrant 3 micro FIFO so that the flow of memory write and non
26 memory write packets can be throttled separately. Once the FIFO(s)
27 become less than half full, a packet is sent to inform the remote
28 FMC that it can resume transmission of one or both types of
29 packets.

30 Figure 10 shows a detailed view of quadrant 3. Quadrant 3
31 processing begins when the first 40 bits of a packet are received
32 over the high speed serial link. Control logic 25 causes the 40
33 bits to be clocked into a staging latch 15. When the rest of the
34 packet arrives, the entire 80 bit packet is clocked into the Rx
35 latch 17. From the Rx latch 17, 64 bits of the packet are moved

1 to a pair of EDC generators/checkers 13a and 13b. The received EDC
2 is compared to that generated and an indication of the outcome is
3 provided to the control logic.

4 The destination of the contents of the packet depend on the
5 packet type bits examined by the control logic and the result of
6 the EDC check. If the EDC check fails, the packet contents are
7 moved to the micro interface FIFO 23. The packet contents are also
8 moved to the micro FIFO 23 if the packet type flags indicate that
9 the packet contains async, interrupt or other control information.
10 Otherwise, the memory address, data and flags are moved to the Rx
11 FIFO 11.

12 Among the 12 packet bits examined by the control logic 25 are
13 the flow control bits, one for throttling memory write packet
14 transmission and the other for throttling non memory write packet
15 transmission. A set flow control bit in a received packet causes
16 the FMC to cease transmission of the corresponding type of packet
17 until a packet is received with that flow control bit reset.

18 Figure 11 shows a block diagram of the AMD 29818 latch used
19 to implement the MC input and output latches 9 and 10 and the Rx
20 latches 15 and 17 and Tx latches 18 and 20 on the FMC. In the FMC
21 design, the normal data path through these latches is the parallel
22 one from the D input through the pipeline register 29 to the Q
23 output. The alternate serial path is accessible by the FMC
24 microprocessor and is primarily used for diagnostic purposes. Note
25 that the mux 30 (which is controlled by the MODE signal) allows
26 either the contents of the shadow register 31 or the D input to
27 serve as the input for the pipeline register 29. Also, note that
28 the output of the pipeline register 29 is feed back into the shadow
29 register 31. Thus, with appropriate hardware control, data can
30 enter the latch in serial and exit in parallel or enter parallel
31 and exit in serial.

32 These features are exploited by the FMC design which allows
33 the microprocessor to serially load a latch in one quadrant, move
34 the data along the normal parallel path from that latch to a latch
35 in another quadrant and then serially read the contents of the

1 destination latch. By comparing the data loaded into the first
2 latch with that read back from the second, the microprocessor can
3 determine if the data path is functional.

4 In the FOMCS architecture, the high speed serial link is
5 implemented with a transmit/receive pair of Gazelle Hot Rod chips.
6 The Gazelle Hot Rod transmitter chip converts 40-bit parallel data
7 into serial data that is transmitted over a fiber or coax link.
8 At the remote end of the link, the data is reconverted to the
9 original parallel data by a Hot Rod receiver chip. Over this link,
10 the FMCs exchange information in the form of 80-bit packets. The
11 Gazelle transmitter sends each 80-bit packet as two 40-bit data
12 frames. When no packet is available to send, the transmitter sends
13 sync frames to maintain link synchronization.

14 The data on the serial data link is 4B/5B NRZI (Non-Return
15 to Zero, Invert on ones) encoded. This allows the receiver to
16 recover both data and clock signals from the data stream,
17 precluding the need for a separate clock signal to be sent along
18 with the data. Data rates as high as 1 Gbps (one billion bits per
19 second) are supported.

20 Transmission error detection is accomplished via an eight
21 bit error detection code (EDC) which is included in each packet.
22 Of the 72 remaining bits in the packet, only 64 are included in the
23 EDC calculation. The eight unprotected bits are the first four
24 bits of each of the 40-bit halves of the packet.

25 The basic packet format is shown in Figure 12.

26 Bits 0 and 40 are used to distinguish the two 40-bit
27 sections of the packet. When receiving a packet, a FMC always
28 expects the first bit of the first 40-bit section to be a zero and
29 the first bit of the second 40-bit section to be a one.

30 The BSD and BSC bits (bits 1 and 41, respectively) are used
31 by the transmitting FMC to throttle transmissions by the FMC
32 receiving the packet. The BSD bit is used to tell the receiving
33 FMC to cease or resume transmission of memory write data packets;
34 the BSC bit is used to tell the receiving FMC to cease or resume
35 transmission of control packets. The bits are set to tell the

1 receiving FMC to cease the associated type of transmissions; they
2 are reset to tell the receiving FMC to resume transmissions.

3 The TYP bit (bit 4) indicates the packet type. If the bit
4 is reset, the packet contains a memory write transfer. If the bit
5 is set, the packet is designated as a control packet. Control
6 packets are used to transfer async data, interrupt data and any
7 other type of information which FMCs must exchange.

8 The VAL bit (bit 5) indicates whether the cross-hatched
9 portions of the packet (bits 6 - 39 and 44 - 71) actually contain
10 any valid information. If the VAL bit is set, the receiving FMC
11 will consider the information in the cross-hatched portions to be
12 valid. If the VAL bit is reset, the receiving FMC will only pay
13 attention to the flow control information encoded in the packet
14 (i.e., the settings of the BSD and BSC bits). When a FMC does not
15 have any data to send, it will periodically transmit packets with
16 the VAL bit reset and the flow control bits set or reset as
17 appropriate.

18 As mentioned earlier, bits 0 - 3 and bits 40 - 43 are not
19 included in the EDC calculation. This means that the BSD bit
20 and/or the BSC bit may be in error in a received packet and no EDC
21 error will be detected. However, even if an invalid flow control
22 indication is received and acted upon, the next packet received
23 will almost certainly correct the problem.

24 The shaded portions of the packet (bits 2 - 3 and bits 42 -
25 43) are reserved. The convention of shading reserved portions of
26 packets is followed throughout this document.

27 Bits 72 - 79 contain the EDC which is computed on bits 4 - 39
28 and bits 44 - 71. The EDC is an 8-bit check byte generated
29 according to a modified Hamming Code which will allow detection
30 of all single- and double-bit errors and some triple-bit errors.
31 Because of the NRZI encoding scheme used by the Gazelle Hot Rod
32 chips, noise on the high speed serial link medium will cause
33 sequential double-bit errors which the receiving FMC will be able
34 to detect by comparing the EDC in the packet to that computed
35 during receipt of the packet.

1 The memory write transfer packet is used by the FMC to
2 transmit a memory address and associated data over the high speed
3 link. The address and data represent a memory write transfer which
4 the FMC received from the MC bus. The format of the packet is
5 shown in Figure 13.

6 The FBT bit (bit 7) is the F-bit associated with the memory
7 write. The F-bit is set if the memory write transfer represents
8 a write to a single byte of memory. The F-bit is reset if the
9 transfer represents a write to a half-word or word.

10 Note that if the memory address is a 24-bit address, bits 44-
11 47 of the packet will contain zeros.

12 The general purpose async data packet format is shown in
13 Figure 14.

14 The packet can contain up to five bytes of async data. The
15 first four bytes are placed in bits 8-39 with the first byte in
16 bits 8-15, the second in 16-23 and so on. The fifth byte is placed
17 in bits 64-71. The receiving FMC determines the actual number of
18 data bytes in the packet by checking the Byte Count field (bits 45-
19 47).

20 The Destination Cluster field (bits 48 - 55) contains the
21 address of the MCS cluster for which the packet is ultimately
22 intended. The FMC in the destination cluster uses the Destination
23 Port field (bits 56 - 63) to determine which general purpose async
24 port to use when transmitting the async data decoded from the
25 packet.

26 The BRK bit (bit 44) indicates that the originating FMC
27 received an async break indication. When the BRK bit is set, the
28 packet does not contain any async data (i.e., the Byte Count field
29 contains a zero). Upon receipt of a general purpose async packet
30 with BRK set, the FMC in the destination cluster flushes its output
31 buffer for the destination port and generates a break on that port.

32 The MCS-II multidrop console data packet format is illustrated
33 in Figure 15.

34 The packet can contain up to four bytes of async data. The
35 four bytes are placed in bits 8 - 39 with the first byte in bits

1 8 - 15, the second in 16 - 23 and so on. The receiving FMC
2 determines the actual number of data bytes in the packet by
3 checking the Byte Count field (bits 45 - 47).

4 The Destination Cluster field (bits 48 - 55) contains the
5 address of the MCS cluster for which the packet is ultimately
6 intended. The contents of this field are only meaningful if the
7 BRD bit is reset.

8 The Source Cluster field (bits 56 - 63) contains the address
9 of the cluster in which the originating FMC resides. Because
10 multiple packets are required to send a multidrop message, a FMC
11 in a cluster may be receiving parts of messages from two or more
12 clusters at the same time. The source cluster field in the packets
13 allows the receiving FMC to segregate message pieces based on
14 source cluster and thus properly reconstruct the messages.

15 The SOM bit (bit 64) indicates whether the async data in the
16 packet represents the start of a multidrop message or not. If the
17 SOM bit is set, the packet contains the first 1 - 4 bytes
18 (depending on the Byte Count) of a multidrop message. If the SOM
19 bit is reset, the packet contains data from within the body of a
20 message.

21 The EOM bit (bit 65) indicates whether the async data in the
22 packet represents the end of a multidrop message or not. If the
23 EOM bit is set, the packet contains the last 1 - 4 bytes (depending
24 on the Byte Count) of a multidrop message. If the EOM bit is
25 reset, at least one more packet of data from the message will
26 follow.

27 The BRD bit (bit 67) is set if the packet is to be broadcast
28 throughout the FOMCS network to all clusters. If the BRD bit is
29 set, all FMCs in a FOMCS hub will forward the packet to their
30 respective remote clusters. All cluster FMCs receiving such a
31 packet will accept it (assuming they have multidrop support
32 enabled).

33 The interrupt packet format is shown in Figure 16.

34 The Destination Cluster field (bits 48 - 55) contains the
35 address of the MCS cluster for which the packet is ultimately

1 intended. The FMC in the destination cluster uses the Destination
2 Line field (bits 56 - 63) to determine which output interrupt line
3 to pulse.

4 A Motorola 68000 microprocessor on the FMC provides the
5 configuration system interface as well as data management for all
6 transfers other than memory write transfers. This includes all
7 asynchronous data, interrupts and error logging. It also provides
8 diagnostic capabilities driven through the configuration
9 programming interface.

10 Four Signetics 68681 DUARTs on the FMC provide support for
11 the exchanges of async serial data between the nodes and/or devices
12 in separate MCS clusters. The async support is designed in such
13 a way that communicating entities need not be aware that the async
14 data is actually traveling over a high-speed fiber or connection.
15 From the perspective of the entities, communication is not
16 functionally different from the case where the nodes and/or devices
17 are directly connected via RS-232 cables. No special protocol
18 information has to be inserted by the communicating entities into
19 the async data stream to allow the data to move through the MCS
20 network. Figure 17 illustrates the handling of async serial data
21 by a FMC in a cluster.

22 As async serial data is received from a node or device, the
23 FMC packetizes the data and transmits the packets over the high
24 speed serial link. In addition to async data, each packet contains
25 an address which facilitates routing of the packet through the MCS
26 network. In the other direction, the FMC receives and decodes
27 packets of async data from the high speed link. The address in the
28 packet is discarded and the async data is passed to the node or
29 device.

30 The address in an async packet is designed to allow routing
31 of the packet through a MCS hub. To accomplish this, the address
32 consists of two fields, a destination cluster field and a
33 destination port field. The destination cluster field indicates
34 the cluster to which the packet is to be routed. The destination
35 port field indicates which async link of the FMC in the destination

1 cluster that the data is intended for. The destination FMC
2 validates the address when it receives the packet. If the
3 destination cluster address does not match the local cluster
4 address, the packet is discarded.

5 The FMC maintains a list of eight async addresses, one for
6 each general purpose async port. When async data is received from
7 one of the eight ports, the FMC looks up the async address for that
8 port and includes it in the packet(s) sent over the high speed
9 link. Thus, there is a static connection between each general
10 purpose async port of a FMC and an async port of a FMC in some
11 other cluster.

12 The contents of the list of async addresses and the local
13 cluster address are established during configuration programming
14 of the FMC. The physical characteristics of the async links (i.e.,
15 baud rate, character length, etc.) are also established during
16 configuration programming.

17 In a MCS hub, the FMCs are programmed to operate in a special
18 async data pass-through mode. This mode of operation is
19 illustrated in Figure 18. Async packets received over the high
20 speed link are decoded by the receiving FMC and sent over the hub
21 MC bus to the other FMCs in the hub. The address field inserted
22 by the FMC at the originating cluster is passed along over the hub
23 MC bus as well as the async data. Each FMC in the hub which
24 receives the address and async data from the MC bus compares the
25 destination cluster field to the address of the cluster at the
26 remote end of its high speed link. If a match occurs, the FMC
27 builds and transmits an async packet over the high speed data link.

28 The remote cluster address used by a FMC in the hub to perform
29 the routing function just mentioned is established during
30 configuration programming. Since the async links of a FMC in a hub
31 are not connected and are ignored by the FMC, no async address list
32 or async link physical characteristics can be programmed.

33 To accommodate the case where the high speed link of a FMC in
34 one hub is directly connected to a FMC in another hub (rather than
35 to a MCS cluster), async address checking may also be disabled in

1 a FMC via configuration programming. Thus, all async packet
2 traffic over the bus of one hub will also appear on the bus of the
3 other hub.

4 The general purpose async ports of a FMC support hardware flow
5 control using DTR and CTS. When a FMC wishes to transmit over a
6 general purpose async port and hardware flow control is enabled
7 for that port, the FMC only transmits as long as it sees CTS
8 asserted. A loss of CTS during transmission causes the port to
9 cease transmission until CTS reappears. Thus, the device connected
10 to an FMC async port can throttle FMC data transmission for that
11 port by asserting and deasserting CTS.

12 The FMC can also throttle a device which is sending it data
13 over a general purpose async port assuming that an appropriate
14 cable is used and the device pays attention to the CTS signal.
15 When the FMC determines that its receive buffer for the port is
16 nearly full, it deasserts DTR to inform the device connected to the
17 port of the condition. Assuming a cable which connects the port's
18 DTR signal to the device's CTS signal, the device will see a loss
19 of CTS. This should cause the device to cease transmission until
20 the FMC asserts DTR to indicate that more data can be accepted.

21 During configuration programming of the FMC, async flow
22 control using DTR and CTS can be enabled or disabled on a per port
23 basis. If DTR/CTS flow control is not desired, FMC general purpose
24 async ports can be configured for XON/XOFF flow control or flow
25 control can be disabled altogether.

26 To deal with the situation in which a FMC is connected to an
27 async data source which is transmitting data much faster than the
28 device connected to the destination FMC can accept it, an async
29 flow control mechanism is also implemented between FMCs. An FMC
30 can send flow control packets to throttle the transmission of async
31 data packets by the FMC connected to the source of the async data.
32 Flow control can be asserted on a per async port basis so that the
33 flow of async packets for other async ports is unaffected.

34 The FMC supports 16 interrupt lines: eight input lines and
35 eight output lines. The lines allow an interrupt pulse generated

1 by a node or device in one MCS cluster to be passed, in effect,
2 across the MCS network to a node or device in another cluster. The
3 pulse enters the FMC in the originating cluster via an input
4 interrupt line, is passed across the MCS network as a packet and
5 leaves the FMC in the destination cluster via an output interrupt
6 line. As with async data, this is designed in such a way that the
7 node need not be aware that the interrupt pulse is travelling over
8 a high speed serial link and not actually directly connected.

9 An overview of interrupt passing for a FOMCS configuration is
10 shown in Figure 19. The process involved in passing an interrupt
11 across FOMCS is very similar to that used to pass async data. The
12 FMC detects the pulse on the input interrupt line and constructs
13 a special interrupt packet which is transmitted over the high speed
14 link. The interrupt packet contains an address which facilitates
15 routing through the FOMCS network. At the destination FMC, the
16 packet is decoded and the address is used to determine which
17 interrupt line should be pulsed.

18 As in the async case, the address in an interrupt packet
19 consists of two fields. The destination cluster field identifies
20 the cluster to which the packet is to be routed. The destination
21 line field indicates which output interrupt line is to be pulsed.
22 As with async packets, the destination FMC validates the cluster
23 field of a received interrupt packet and discards the packet if the
24 destination cluster does not match the local cluster address.

25 The FMC maintains a list of eight interrupt line addresses,
26 one for each input interrupt line. When the FMC detects a pulse
27 on one of the input interrupt lines, it looks up the line address
28 and includes it in the packet sent over the high speed link. Thus,
29 there is a static connection between each input interrupt line of
30 a FMC and an output line of a FMC in some other cluster. The
31 contents of the list of interrupt line addresses are established
32 during configuration programming of the FMC.

33 In a FOMCS hub, the FMCs are programmed to operate in
34 pass-through mode very similar to that described earlier for async
35 data. Interrupt packets received over high speed link are

1 distributed over the MC bus to the other FMCs in the hub. Each FMC
2 in the hub receives the interrupt line address from the MC bus and
3 compares the cluster field to the address of the cluster at the
4 remote end of its high speed link. If a match occurs, the FMC
5 builds and transmits an interrupt packet over the high speed link.

6 The remote cluster address used by a FMC in the hub to perform
7 the routing function just mentioned is established during
8 configuration programming. Interrupt address checking can also be
9 disabled in a FMC to accommodate the situation where hubs are
10 directly connected. Since the interrupt lines of a FMC in a hub
11 are not connected and are ignored by the FMC, no input interrupt
12 address list can be programmed.

13 One channel of a Signetics 68681 DUART on the FMC is used to
14 provide support for the MCS-II multidrop console link. (The other
15 channel of this DUART is used for the configuration programming
16 link.) The FMC behaves basically as a gateway to allow console
17 traffic to flow between nodes in separate MCS-II clusters. In each
18 cluster, the FMC intercepts messages intended for remote nodes and
19 transmits them as packets to the FMCs in the appropriate clusters.
20 The FMCs receiving the packets decode them and send the messages
21 to the destination nodes. The fact that this is happening is
22 essentially transparent to the nodes. An example of a FOMCS
23 network linking clusters of MCS-II nodes appears in Figure 20.

24 In the MCS-II multidrop console environment, one of the drops
25 acts as the master and all others are slaves. Only the master can
26 initiate message exchanges on the multidrop link; a slave can only
27 transmit after receiving a poll message from the master. The FMC
28 in a cluster assumes that there is a local master in the cluster.
29 When the FMC receives a message addressed to a remote node, it
30 transmits the message as one or more packets over its high speed
31 serial link; when the FMC receives one or more packets from the
32 high speed link containing a message from a remote node, it buffers
33 the message until it is polled by the master. Then, the FMC
34 transmits the saved message over the local multidrop link. Note
35 that a cluster FMC's slave address on the multidrop link is its MC

1 bus node id.

2 The handling of multidrop message data in the FOMCS network
3 is basically the same as that for general purpose async data and
4 interrupts. Multidrop data is routed through the FOMCS network in
5 special packets. Each multidrop data packet contains a destination
6 cluster address (which is validated by the destination FMC) and
7 part of a message. Unlike interrupt packets or packets of general
8 purpose async data, multidrop message packets also contain a source
9 cluster address. The receiving FMC uses the source address to
10 segregate pieces of messages from different clusters into different
11 buffers.

12 The source cluster address which a node inserts into a
13 message is supplied by the FMC in the cluster. The multidrop
14 master periodically sends a request-cluster-address message to the
15 FMC which causes it to broadcast a response message containing the
16 cluster address to all nodes in the cluster. This cluster address
17 is the address established during configuration programming via the
18 Define Cluster Address command. Note that in a MCS-II cluster
19 where there is no FMC to supply the source cluster
20 communication between nodes is unaffected because the source
21 cluster address field is not included in messages exchanged by
22 nodes in the same cluster.

23 In FOMCS configurations where a hub is present, multidrop
24 data is routed from the receiving FMC to the other FMCs in the hub
25 via the hub MC bus. Packets are then built and transmitted from
26 FMCs in the hub to the FMCs in the destination clusters. As with
27 general purpose async and interrupts, a FMC in the hub looks at the
28 destination cluster address to determine if it should build and
29 transmit a packet over its high speed link. If the destination
30 cluster matches the address of the cluster connected to the FMC's
31 serial data link, the FMC forwards the multidrop data.

32 The MCS-II multidrop broadcast capability is also supported
33 by FOMCS. When the FMC in a cluster receives a message over the
34 local multidrop link that is to be broadcast to all nodes, it sends
35 the message across its high speed link in packets which have a

1 broadcast flag set. FMCs in a hub will always forward such packets
2 regardless of the contents of the destination cluster field. All
3 cluster FMCs receiving the broadcast message will transmit it over
4 their local multidrop links when polled by the local masters.

5 During configuration programming, a cluster FMC is programmed
6 with a list of remote cluster addresses. When multidrop support
7 is enabled, the FMC intercepts any message on the multidrop link
8 which is addressed to a node in one of the clusters in the list.
9 The local multidrop master can obtain the list from the FMC by
10 sending it an inquiry message. The FMC responds by encoding the
11 list into a remote-cluster-list message which it sends over the
12 multidrop link.

13 The FMC supports an optional configuration which allows
14 automatic failover to a secondary high speed serial link should the
15 primary link go down. This high availability feature is
16 illustrated in Figure 21.

17 The secondary FMCs in each cluster monitor the health of the
18 primary FMCs. Each secondary FMC is configured for the same MC bus
19 node id as the primary it monitors but the secondaries do not
20 interact on the MC bus as long as the primaries are healthy. In
21 other words, all memory write traffic between the clusters goes
22 over the primary high speed link until a failure occurs. Likewise,
23 all other types of packet traffic (i.e., async, interrupt, etc.)
24 go over the primary link until a failure is detected.

25 To keep the secondary FMCs informed as to their health, the
26 primaries periodically send their respective secondaries an "I'm
27 okay" indication via a health-check link. The UART of a Motorola
28 68901 MFP on the FMC is used to implement the health-check link.
29 The primaries also periodically exchange test packets to determine
30 if the primary link is still functioning.

31 If a secondary FMC does not receive an "I'm okay" indication
32 within a specified time period since the last indication or if the
33 secondary receives a link failure indication from the primary, the
34 secondary initiates the fail-over process by: 1) forcing the
35 primary FMC to cease operation, and 2) sending a failure-detected

1 packet to the secondary in the remote cluster. The remote
2 secondary FMC then forces its primary FMC to cease operation. The
3 remote secondary then sends a fail-over-complete packet back to
4 the secondary which detected the failure. Subsequent communication
5 between the clusters occurs over the secondary high speed link.

6 A secondary FMC forces a primary to cease operation by
7 asserting a special additional signal which is included in the
8 health-check link cable connecting the primary and secondary. When
9 asserted this signal places the primary FMC into a
10 hardware-controlled offline state. In this state, all I/O
11 interfaces on the FMC are disabled so that the FMC is unable to
12 assert any signals on the MC bus, transmit over the high speed
13 link, transmit over any of its async links or pulse any of its
14 output interrupt lines. The effect is identical to that achieved
15 by putting the FMC online/offline switch into the offline position.

16 The (old) primary FMC is held in this offline state until the
17 secondary FMC (which is the new primary) is reset via a power
18 cycle, hardware reset or a received Reset command.

19 While the old primary FMC is being held in this offline state,
20 it monitors the health-check link (if it is operational enough to
21 do so) for the receipt of "I'm okay" characters from the new
22 primary. If the old primary receives such characters, it
23 reconfigures itself to assume the secondary role. Thus, when an
24 attempt is made to return the old primary FMC to an online state,
25 it behaves as a secondary and remains in a (firmware-controlled)
26 offline state. This prevents the old primary from contending with
27 the new primary on the MC bus.

28 The fail-over process can also be initiated manually via
29 the configuration programming link. To cause a fail-over, a
30 command is sent over the programming link to one of the secondary
31 FMCs. The fail-over proceeds as described above and when it is
32 complete, a response indicating such is returned via the
33 programming link.

34 While the previous discussion relates to high availability
35 of the high speed link between clusters, the high availability

1 feature can be extended to the hub in a FOMCS star network. High
2 availability in a hub is achieved by having a secondary FMC for
3 each primary FMC in the hub. Thus, each cluster is connected to
4 the hub by two high speed links, a primary and a secondary.
5 Fail-over to a secondary link is essentially the same as the
6 cluster to cluster case.

7 Note that to really achieve high availability in a cluster,
8 nodes which have async or interrupt connections to the primary
9 FMC must have a redundant set of connections to the secondary.
10 When the secondary FMC takes over, the nodes must switch over to
11 the secondary async and interrupt connections. To facilitate this
12 process, one of the output interrupt lines of a secondary FMC can
13 be used to inform the node(s) that fail-over has occurred. During
14 configuration programming of a secondary FMC, an output interrupt
15 line can be designated for this purpose.

16 The fail-over process is not automatically reversible.
17 Once the secondary FMCs have taken over, they become in effect the
18 primaries. When the previous primary FMCs and/or high speed link
19 are repaired, they must be programmed to behave as the secondaries
20 (if they have not already reconfigured themselves as secondaries).

21 Configuration of a FMC as a primary or secondary occurs
22 during configuration programming. The maximum time period that a
23 secondary will allow between "I'm okay" indications from the
24 primary is also established. The high availability feature can
25 also be disabled altogether.

26 Another usage of the microprocessor is to allow diagnosis
27 of an individual FMC and/or high speed serial link without
28 bringing the entire MCS network offline. This capability is
29 particularly useful in the star network environment where it is
30 undesirable to shut down the entire hub just to diagnose a problem
31 between the hub and a particular cluster. If the problem turns out
32 to be the high speed link or the FMC in the cluster, it can be
33 corrected without taking down the entire hub. Of course, software
34 resynchronization will probably be necessary between the affected
35 cluster and the other clusters it was communicating with but the

1 rest of the clusters connected to the hub can continue to
2 communicate without interruption.

3 Special operational modes, referred to as hub diagnostic and
4 cluster diagnostic modes, are available to allow the FMC to be
5 isolated from the high speed coax or fiber link for testing
6 purposes. In these modes, the FMC operation is identical to normal
7 hub or cluster mode except that the output of the Gazelle Hot Rod
8 Tx chip is directly connected to the input of the Rx chip. To
9 configure the FMC to operate in hub or cluster diagnostic mode, the
10 Set FMC Online/Offline command used to place the FMC in a
11 firmware-controlled offline state. Then, the desired mode is
12 selected via the Set Operational Mode command and the FMC is
13 returned to an online state by means of a second Set FMC
14 Online/Offline command.

15 Individual functional areas of the FMC can be tested by
16 sending Specify Diagnostic Loopback Data commands to the FMC.
17 Variations of the Specify Diagnostic Loopback Data command can be
18 used to invoke memory write transfer loopback, async data loopback
19 and interrupt loopback. A FMC will only accept such commands when
20 in the firmware-controlled offline state.

21 The FMC supports three diagnostic loopback modes for memory
22 write transfers: internal, external and MC bus loopback. Internal
23 loopback loops data through the FMC from the MC bus input latch to
24 the MC bus output latch. External loopback tests the same path
25 except that the data is actually transmitted over the high speed
26 serial link and looped back via an external loopback cable
27 connection. Internal loopback mode is shown in Figure 22.
28 External loopback mode is shown in Figure 23. MC bus loopback
29 loops data from the MC bus output latch to the MC bus input latch
30 via the MC bus. This loopback mode is illustrated in Figure 24.

31 Note that when internal or external loopback is performed,
32 the FMC hit/translation RAM must also be programmed appropriately
33 to achieve the desired effect. Loopback modes can be used to
34 specifically test reflection region or address translation logic
35 or the hit/translation RAM can be programmed such that all regions

1 are reflected and address translation does not actually do anything
2 (i.e., the original address and the translated address are the
3 same).

4 In internal loopback mode, the MC bus interface of the FMC
5 is disabled and the Gazelle Hot Rod chips are configured such that
6 the serial output of the transmitter chip is connected directly to
7 the serial input of the receiver chip. The Specify Diagnostic
8 Loopback Data command which initiated the loopback also contains
9 an address and data pattern to be looped through the FMC hardware.
10 The FMC inserts the address and data pattern into the MC bus input
11 latch. The address and data proceed through the Tx path and at
12 the end of the Tx path are transmitted serially in a packet. The
13 Packet is looped back through the Rx path and the address and data
14 end up in the MC bus output latch. The FMC removes the address and
15 data from the latch and includes them in the response sent back
16 via the programming link.

17 In external loopback mode, the MC bus interface of the FMC
18 is disabled. Again, the Specify Diagnostic Loopback Data command
19 which initiated the loopback contains an address and data pattern
20 to be looped through the FMC hardware and the external loopback
21 connection. The FMC inserts the address and data pattern into the
22 MC bus input latch. The address and data proceed through the Tx
23 path and at the end of the Tx path are transmitted serially in a
24 packet. The packet is looped through the external loopback
25 connection, received again by the FMC and moves through the Rx path
26 to the MC bus output latch. A response is sent back via the
27 programming link containing the address and data pattern read from
28 the latch.

29 In MC bus loopback mode, the high speed serial link interface
30 is disabled and the MC bus interface of the FMC is enabled but
31 functions somewhat differently from normal. To perform the MC bus
32 loopback, the FMC inserts the address and data pattern from the
33 Specify Diagnostic Loopback Data command into the MC bus output
34 latch. The FMC MC bus interface hardware then requests the bus and
35 when it receives a grant, generates a transfer on the MC bus with

1 the data valid signal deasserted. This causes any nodes on the bus
2 to ignore the transfer. The FMC MC bus interface hardware,
3 however, is configured to receive its own transfer. The looped
4 data is read from the MC bus input latch and returned in the
5 response sent back via the programming link.

6 The FMC supports three diagnostic loopback modes for general
7 purpose async data: internal, external, and port-port. The three
8 varieties are illustrated in Figure 25. The FMC performs the
9 requested async loopback as the result of a Specify Diagnostic
10 Loopback Data command received over the configuration programming
11 link.

12 The Specify Diagnostic Loopback Data command informs the FMC
13 as to which async port(s) will participate in the loopback test and
14 provides the data to be looped. If an async port is selected for
15 internal loopback, the FMC initializes the port hardware to wrap
16 transmitted data back to the receive side of the port. The FMC
17 then causes the port to transmit the data. If the port is
18 functioning correctly, the FMC immediately receives the data back
19 again. The received data is passed back over the programming link
20 in the command response. If no data is received, the FMC returns
21 a response indicating such.

22 External loopback requires that a loopback plug be connected
23 to the port which wraps transmitted data back to the receive pin
24 of the port. The FMC initializes the port hardware for normal
25 operation and transmits the data via the specified port. Again,
26 the data should be immediately received from the same port. The
27 response to the command contains the received data or indicates
28 failure of the loopback operation.

29 Port-port loopback requires that a RS-232 cable connect the
30 two selected ports. The FMC initializes the ports for normal
31 operation and transmits the data via the port specified in the
32 Specify Diagnostic Loopback Data command. The response to the
33 command contains the data received from the other port or indicates
34 failure of the loopback operation.

35 The FMC supports external loopback for the interrupt lines.

1 A Specify Diagnostic Loopback Data command to the FMC selects a
2 pair of interrupt lines that will participate in the loopback, one
3 input and one output. Note that loopback requires that a wire
4 connect the selected interrupt lines. To perform the loopback,
5 the FMC generates a pulse on the output line of the pair and
6 reports in its command response whether or not a pulse was detected
7 on the input line.

8 The FMC supports two diagnostic loopback modes for the MCS-II
9 multidrop console link: internal and external. Internal loopback
10 is performed exactly as with the general purpose async ports.
11 External loopback is functionally identical to the method used with
12 the general purpose async ports, but because of the unique nature
13 of the multidrop link, no external loopback cable is required.
14 However, the external loopback test will drive the multidrop link.
15 Therefore, if testing is desired without bringing the entire
16 multidrop link offline, the MCS cable which carries the multidrop
17 link should be unplugged from the FMC chassis backplane prior to
18 initiating diagnostic tests and replaced when testing is concluded.

19 The FMC supports two diagnostic loopback modes for the high
20 availability health-check link: internal and external. Both modes
21 are performed in the same manner as with the general purpose async
22 ports.

23 Each packet transferred across the high speed serial link in
24 a FOMCS configuration contains an error detection code (EDC) which
25 allows the receiving FMC to determine if an error occurred during
26 transmission. Packets received with good EDC are also checked for
27 illegal or illogical bit settings. Parity is checked for transfers
28 received across the MC bus and transfers received with good parity
29 are also checked for illegal or illogical bit settings.

30 For packet errors and transfer errors detected by the
31 receiving FMC, the FOMCS philosophy is to report the error to the
32 node(s) in the nearest cluster rather than reporting the error to
33 all nodes in the network. No attempt is made to handle errors in
34 memory write transfer packets differently from errors detected in
35 other types of packets because once an error is detected in a

1 packet, the entire contents of the packet are suspect including the
2 packet type field.

3 The method of reporting errors is illustrated in Figures 26
4 and 27. When a FMC in a cluster detects an error in a packet
5 received over the high speed link or detects an error in a MC bus
6 transfer, it reports the error to the node(s) in the cluster by:
7 1) forcing a parity error on the MC bus and/or 2) directly
8 interrupting the node(s). To force a parity error, the FMC
9 arbitrates for the MC bus and when granted access, generates a
10 transfer where the parity driven on the bus does not match the
11 address and data. This causes the MC port of the each node to
12 detect a transfer with bad parity and (hopefully) report it to the
13 node via a parity error interrupt. The direct interrupt approach
14 utilizes one or more of the FMC's eight output interrupt lines.

15 When a FMC in a hub detects an error in a transfer received
16 over the hub MC bus or detects an error in a packet received over
17 its high speed link, it builds an error packet and transmits it to
18 the FMC at the other end of the high speed link. Receipt of the
19 error packet causes the FMC in the cluster to report the error as
20 described earlier. Note that whether or not the FMC in the cluster
21 receives the packet correctly, an error will be reported.

22 To assist with diagnosis of packet transmission or MC bus
23 transfer errors, the FMC keeps a copy of the last bad packet (if
24 any) and the last bad MC bus transfer (if any) that it received.
25 These copies can be accessed at any time via the configuration
26 programming link.

27 During configuration programming, the error reporting
28 behavior of a FMC in a cluster is established. The FMC can be
29 programmed to report errors by generating MC bus parity errors
30 and/or by generating interrupts. If the interrupt approach is
31 selected, one or more output interrupt lines of the FMC can be
32 programmed as outputs for error signals. Each of the selected
33 lines can be further qualified as to when the line is pulsed: 1)
34 when a bad MC bus transfer is detected (or when an error packet is
35 received indicating that the remote FMC detected a bad transfer),

1 2) when a bad packet is received over the high speed link (or when
2 an error packet is received indicating that the remote FMC detected
3 a bad packet), or 3) when either a bad transfer or a bad packet is
4 detected.

5 Although the present invention has been shown and described
6 with reference to preferred embodiments, changes and modifications
7 are possible to those skilled in the art which do not depart from
8 the spirit and contemplation of the inventive concepts taught
9 herein. Such are deemed to fall within the purview of the
10 invention as claimed.

We claim:

1. A system comprising:

a first and second set of plurality of nodes;

a first data bus associated with and connecting said first set of plurality of nodes;

a second data bus associated with and connecting said second set of plurality of nodes;

each node of said sets of plurality of nodes including a processing unit, a memory, a bus coupled to the processing unit and memory, and a sensor means for sensing a write to the memory and for transmitting the sensed write on said associated data bus;

first converter means connected to the first data bus for converting data on said first data bus to corresponding optical signals and received optical signals to corresponding data for transmission on said first data bus;

second converter means connected to the second data bus for converting data on said second data bus to corresponding optical signals and received optical signals to corresponding data for transmission on said second data bus; and

fiber optic means for optically transmitting data from one converter means to the other converter means.

2. The system of claim 1 wherein each said converter means includes conversion means for converting parallel data to serial optical signals and vice versa.

3. The system of claim 1 wherein each said converter means includes a FIFO for temporarily storing data.

4. The system of claim 1 wherein each node further includes I/O means for introducing I/O data into memory and wherein said sensor means responsive to a write to memory of I/O data transmits same on said associated data bus.

5. A system for connecting memory coupled systems, comprising:

a plurality of nodes;
a first data bus connecting a first group of said plurality of nodes;
a second data bus connecting a second group of said plurality of nodes;
first converter means connected to the first data bus;
second converter means connected to the second data bus;
first fiber optic means for carrying transmitting data from the first converter means to the second converter means;
second fiber optic means for carrying transmitted data from the second converter means to the first converter means;
the first and second converter means each comprising:
input latch means for receiving data from a respective data bus;
hit and translation RAM means connected to the input latch means for determining the destination of data received from the data bus;
first micro-interface means connected to the input latch means for controlling the hit and translation RAM means;
transmission FIFO means connected to the input latch means for latching the data received from the data bus;
error detection means for determining if an error exists in the data in the transmission FIFO means;
first and second transmission latches connected to the transmission FIFO means;
transmitter means connected to the first and second transmission latches for transmitting the data to another converter means;
receiver means for receiving data transmitted from another converter means;
first and second receiver latches for latching the received data;
error detection means connected to the first and second latch means for checking if an error exists in the received data;
second micro interface means for testing the received

data if the check by the error detection means fails;

receive FIFO means connected to the first and second receive latch means for holding the received data after checking by the error detection means;

output latch means for transmitting the received data to the respective data bus.

6. A system as claimed in Claim 5, further comprising first and second backup controllers for transmitting data between memory coupled systems upon a determination the first and second controllers are not working properly.

7. A system as claimed in Claim 6, wherein said input, output, transmit and receive latches of each of said plurality of controllers are able to be accessed in both a parallel and serial fashion.

8. A system as claimed in claim 1, wherein data is transmitted through the optical fiber means in 80 bit data frames.

9. A method comprising the steps of:

establishing first and second data links;

establishing first and second sets of nodes, each node including a processing unit, a memory, a bus coupled to the processing unit and memory and a sensing means for sensing a write to memory;

sensing a write to a memory in one of said nodes of said first set;

transmitting said sensed write on said first data link; and

sensing the sensed write transmitted on said first data link and optically transmitting same to a remote point whereupon it is transmitted on said second data link to the memory of one of the nodes of said second set without intervention of the processing unit of said one node of said second set.

10. The method of claim 9 comprising the further steps of:
writing I/O data from an I/O source into the memory of a node
in one of the sets of nodes;
sensing the written I/O data and transmitting same on the data
link associated with said node; and
optically transmitting the written I/O data to a memory in a
node of the other sets of nodes via its associated data link.

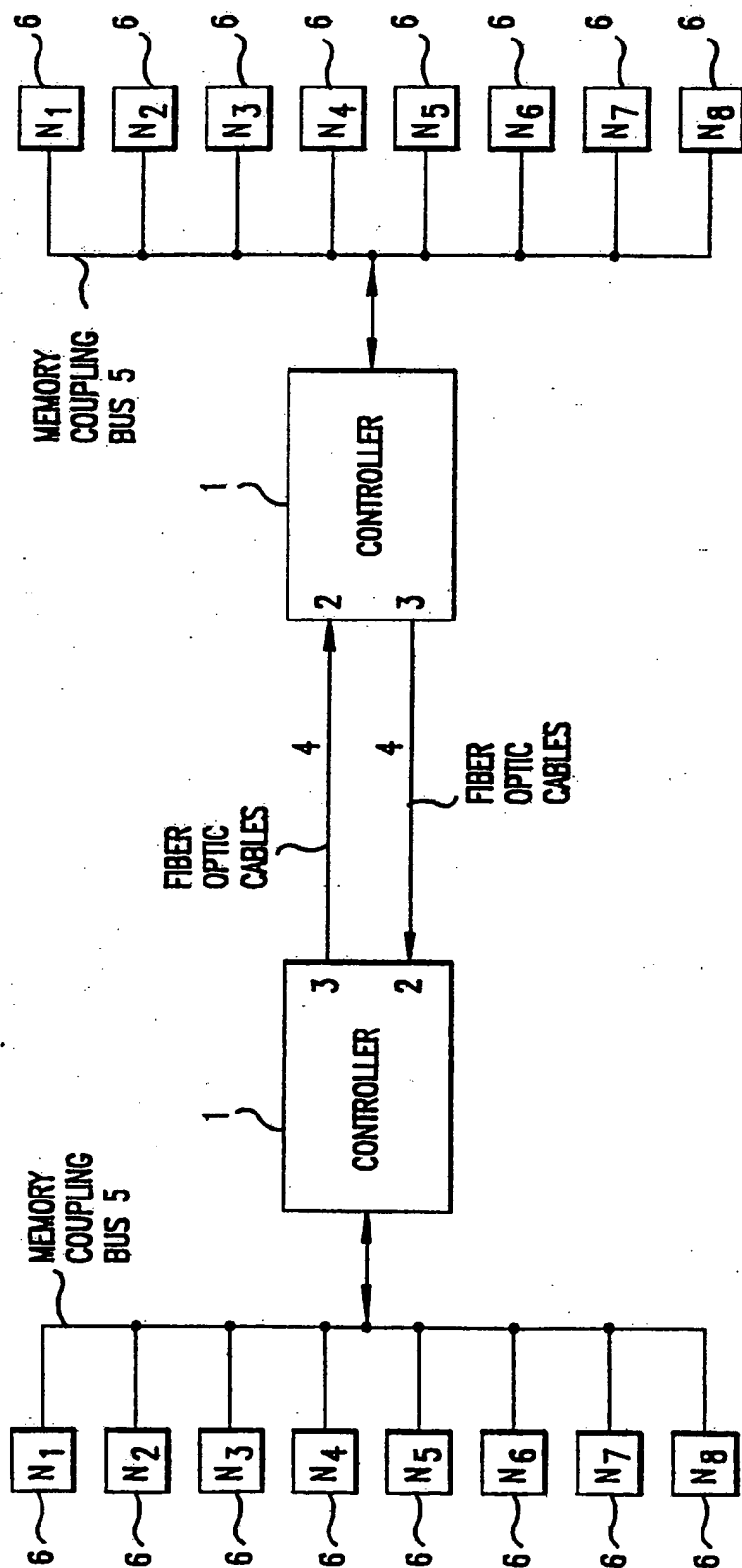


FIG. 1

2 / 31

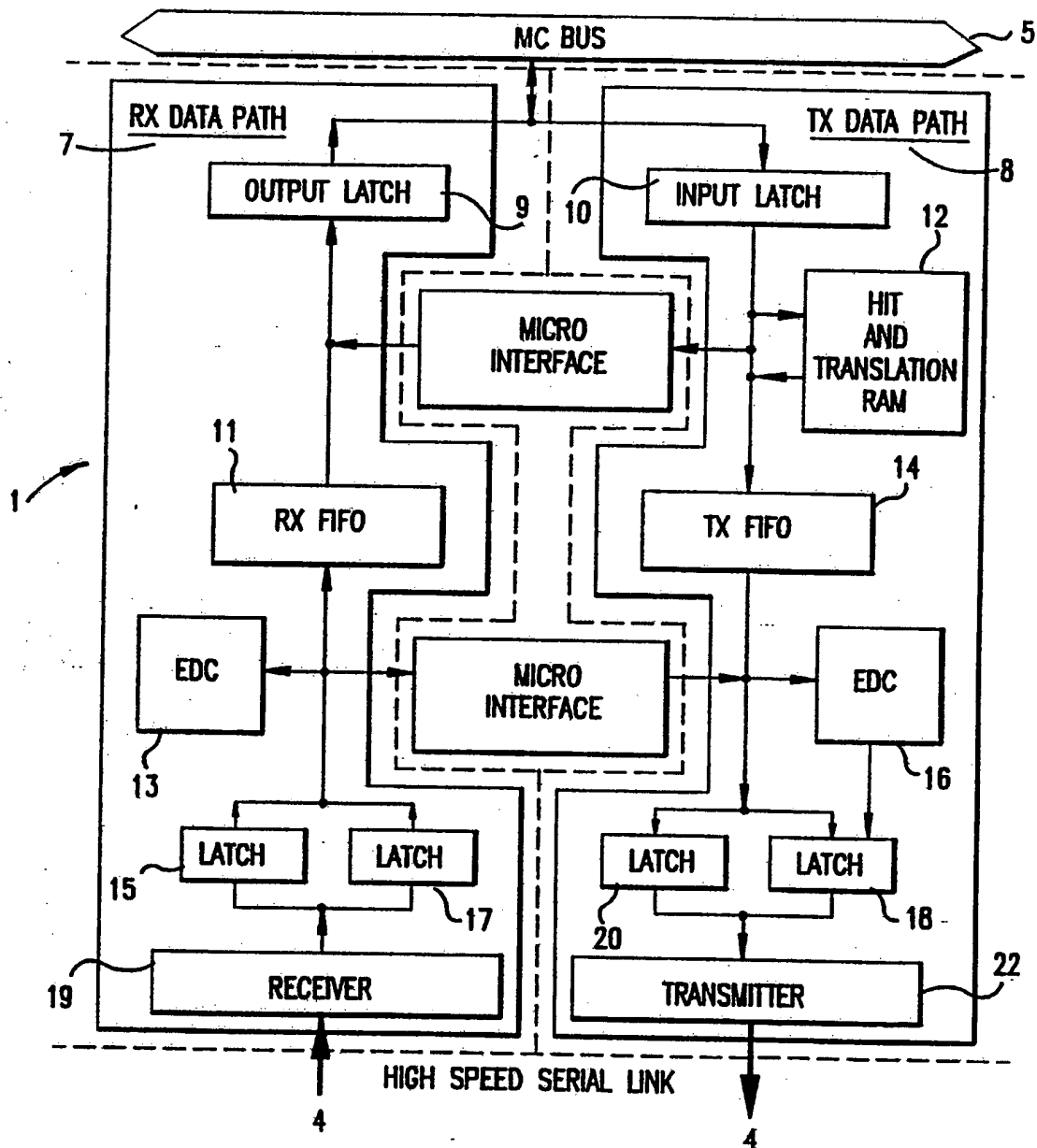


FIG.2

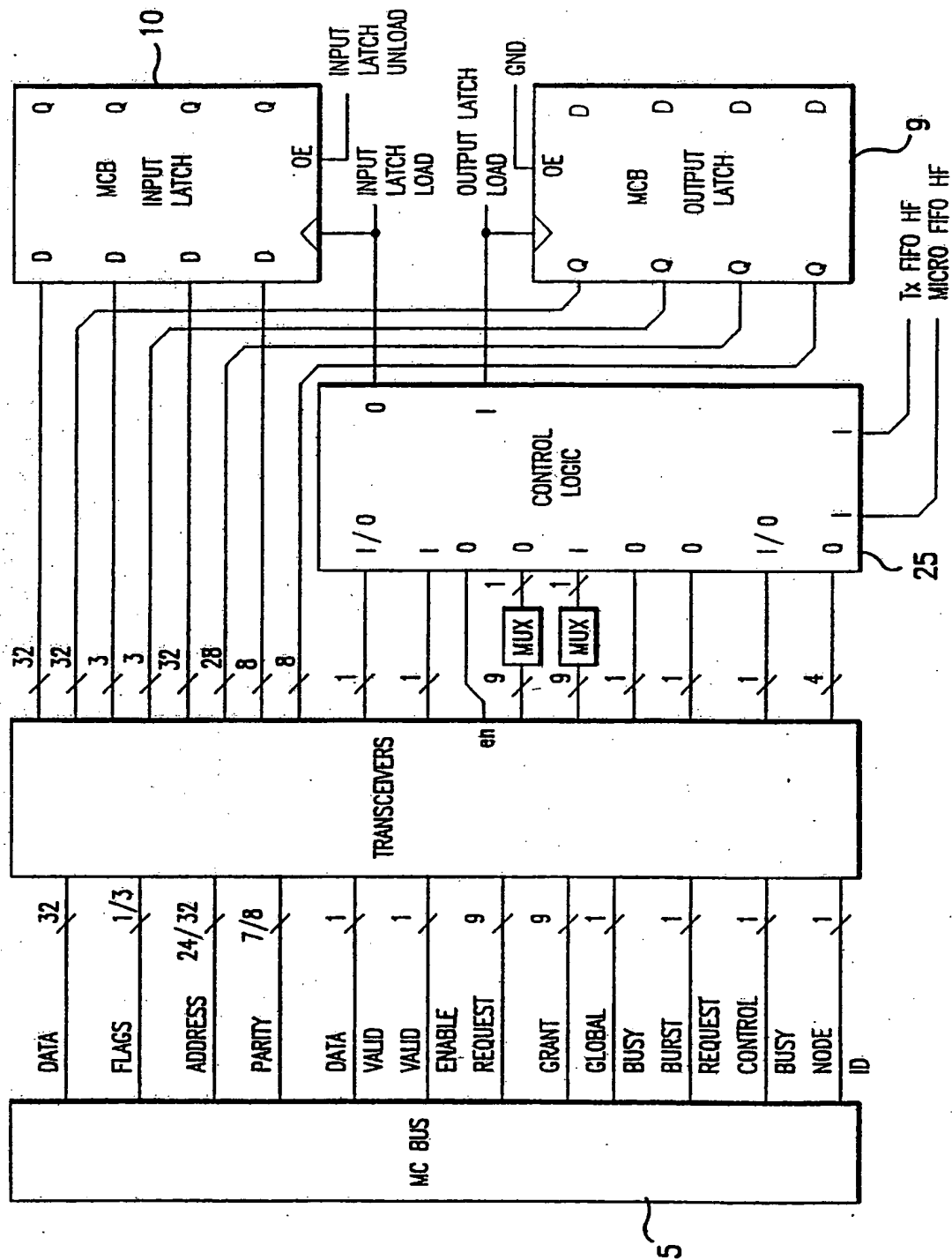


FIG.3

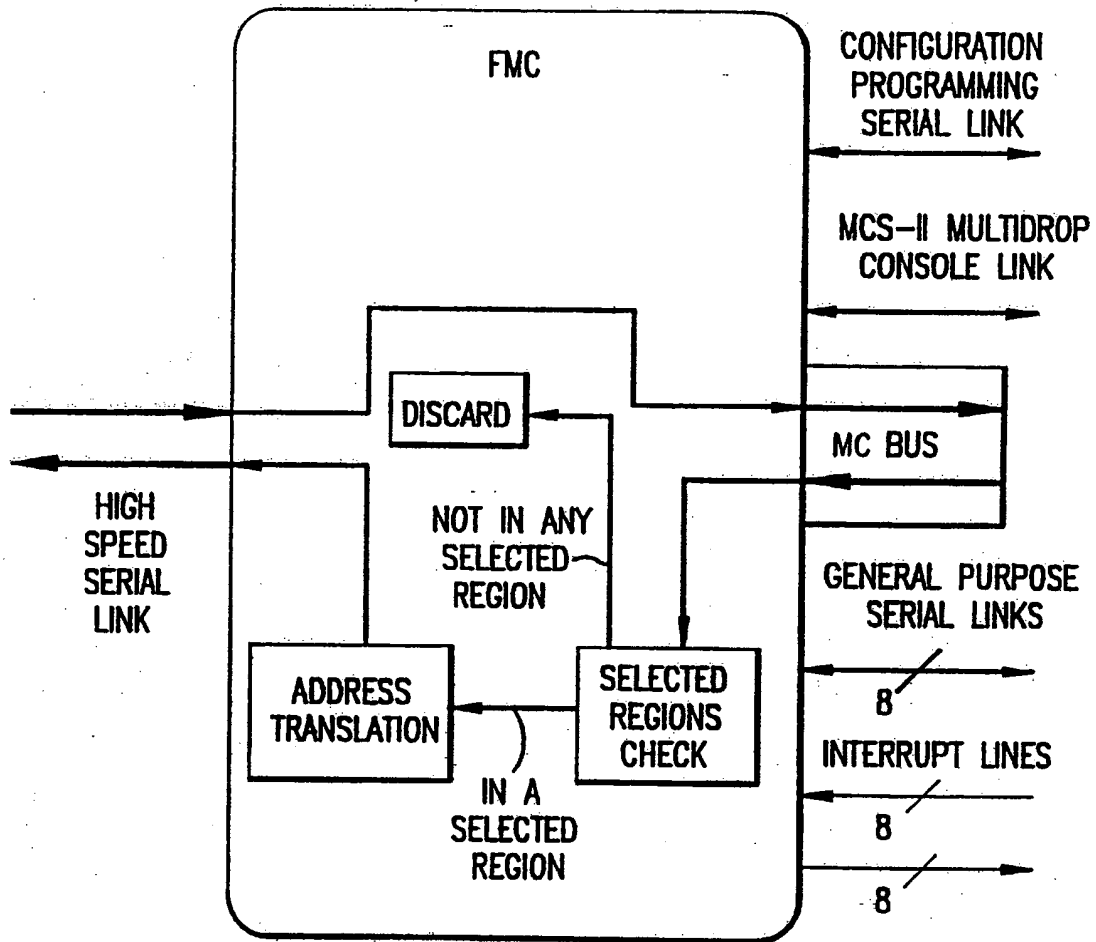
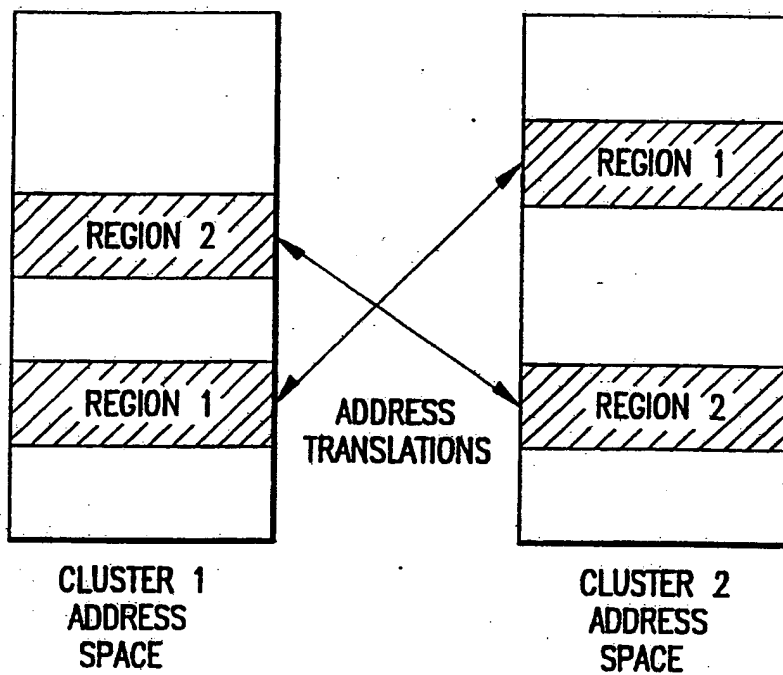


FIG.4



(NOTE THAT ONLY THE SHADED
REGIONS IN EACH CLUSTER ARE
REFLECTED OVER FIBER.)

FIG.5

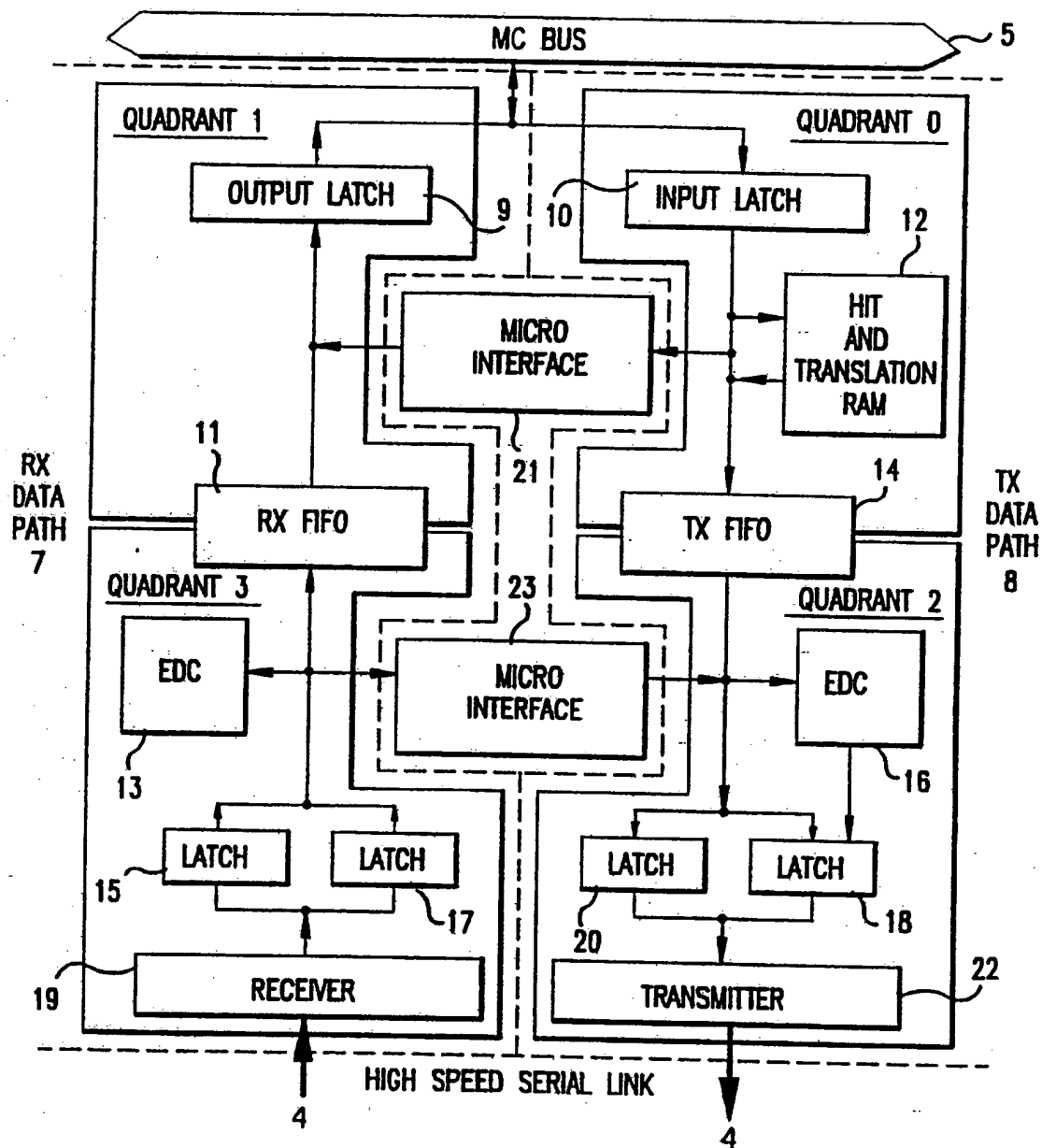


FIG. 6

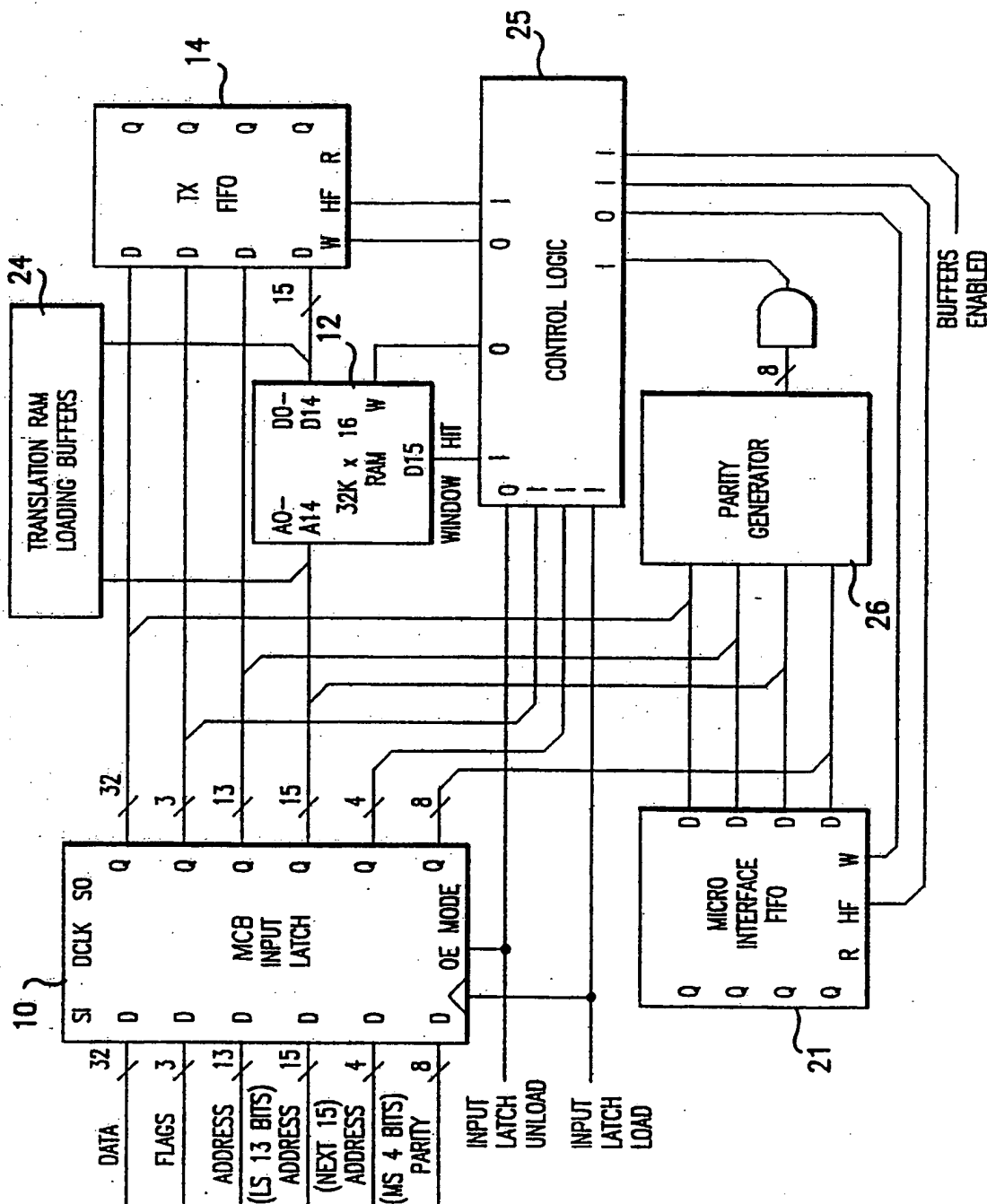


FIG. 7

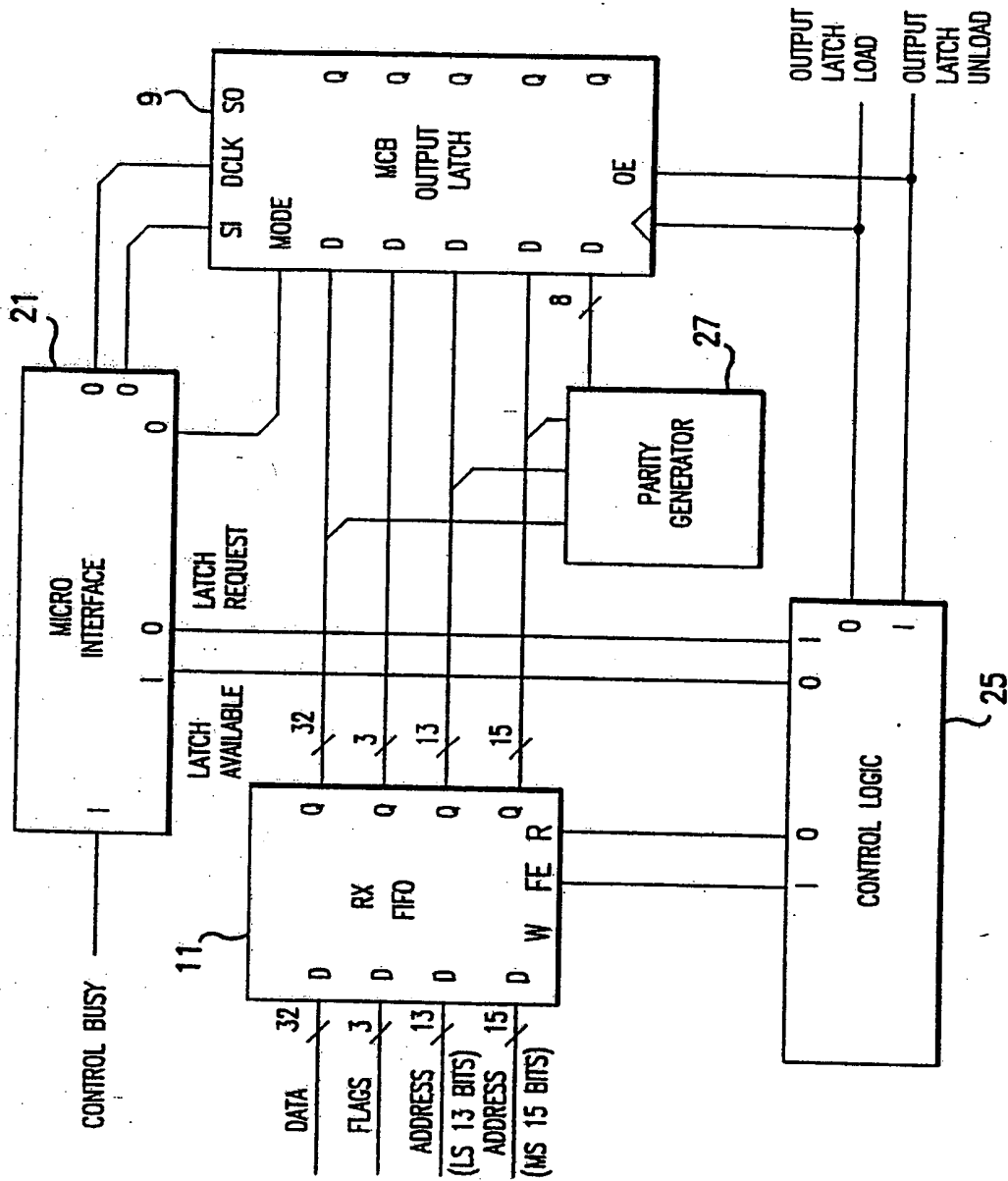
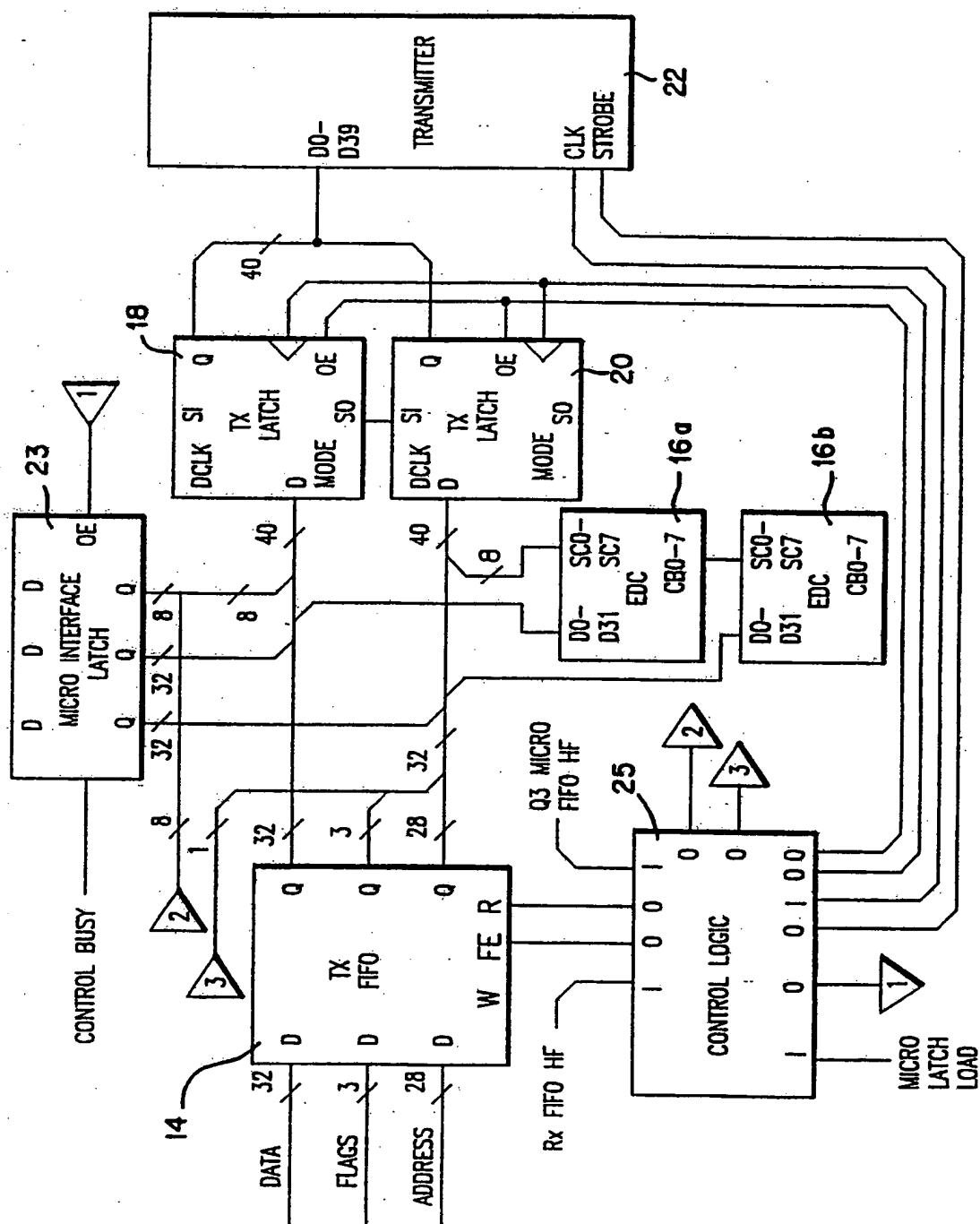
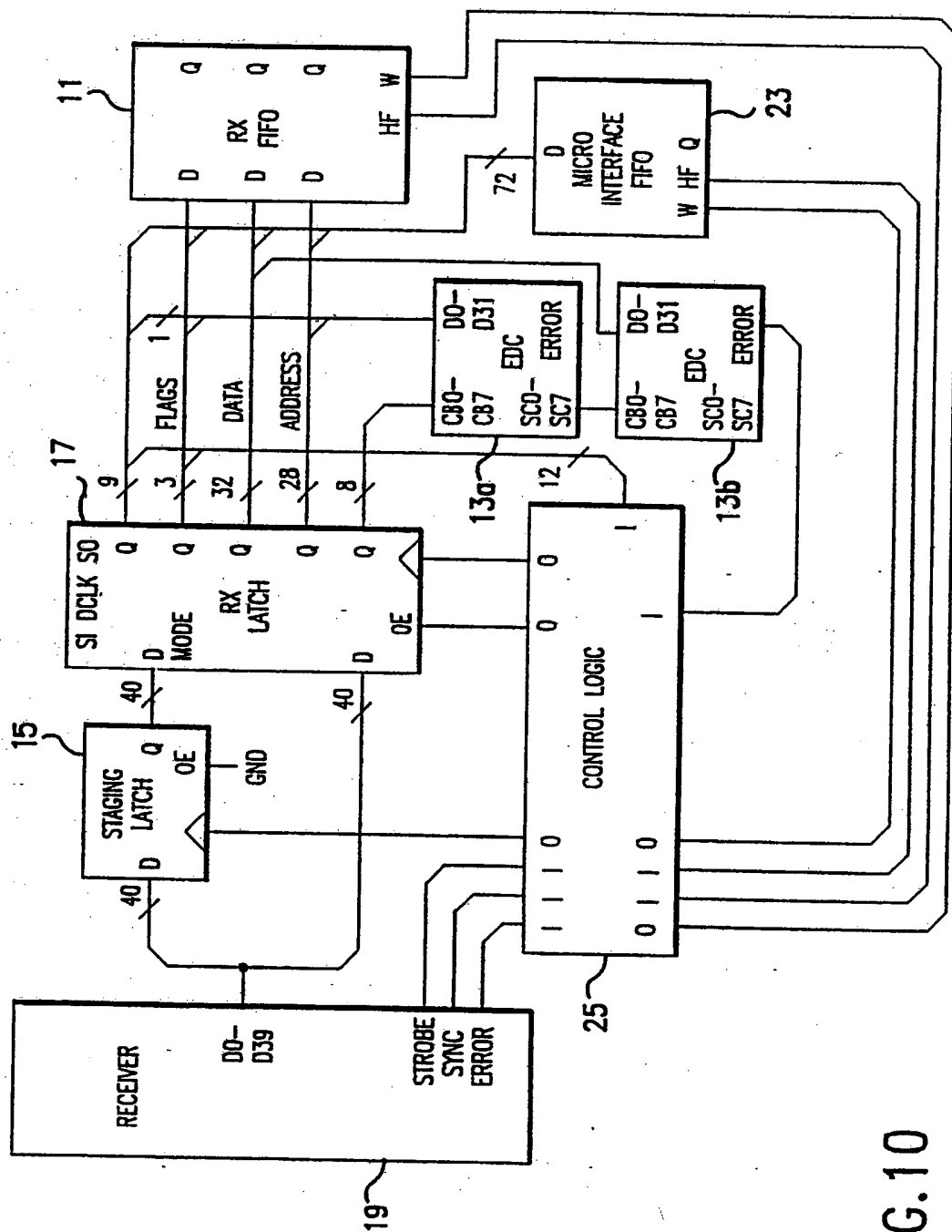


FIG. 8





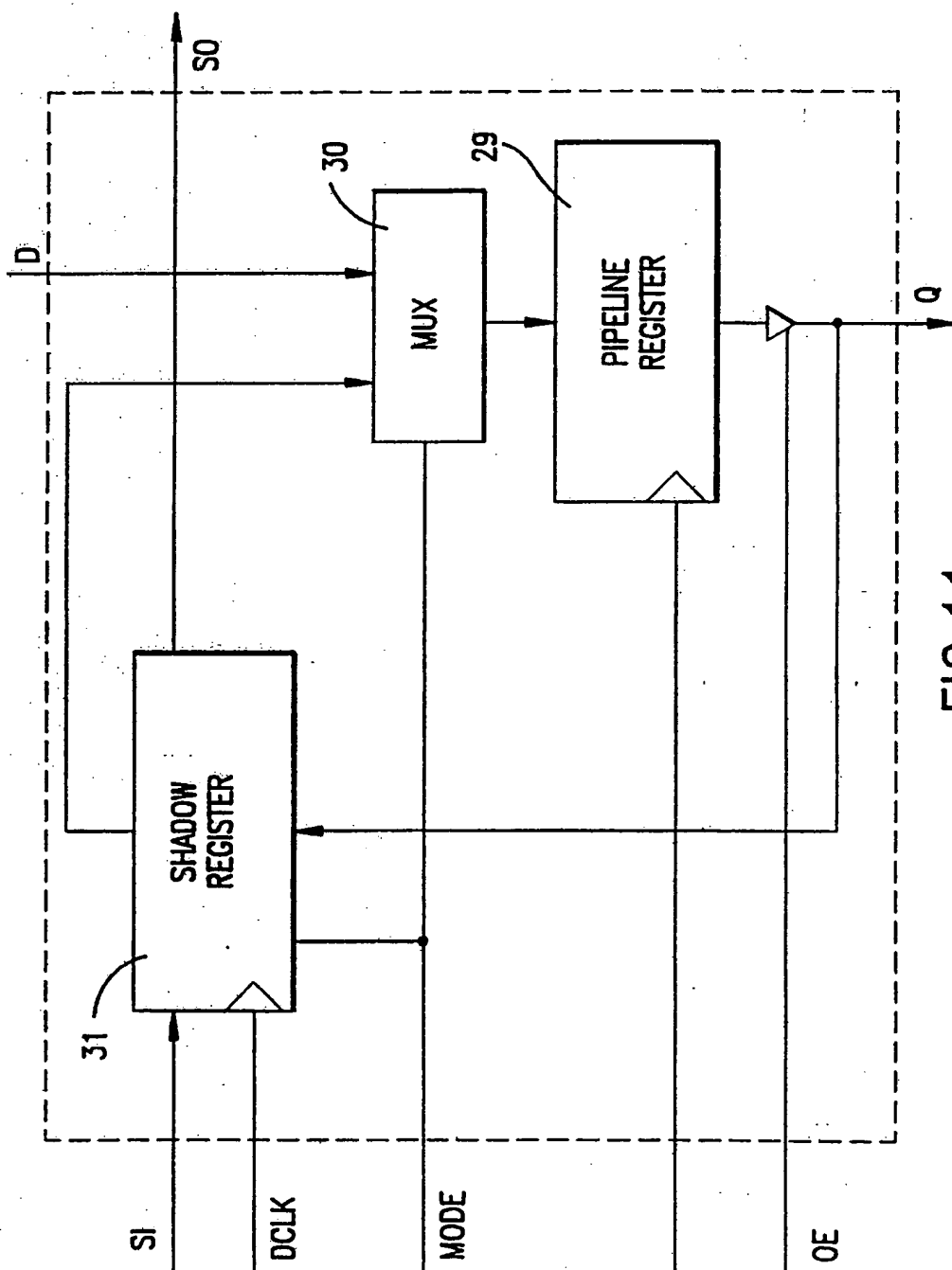


FIG. 11

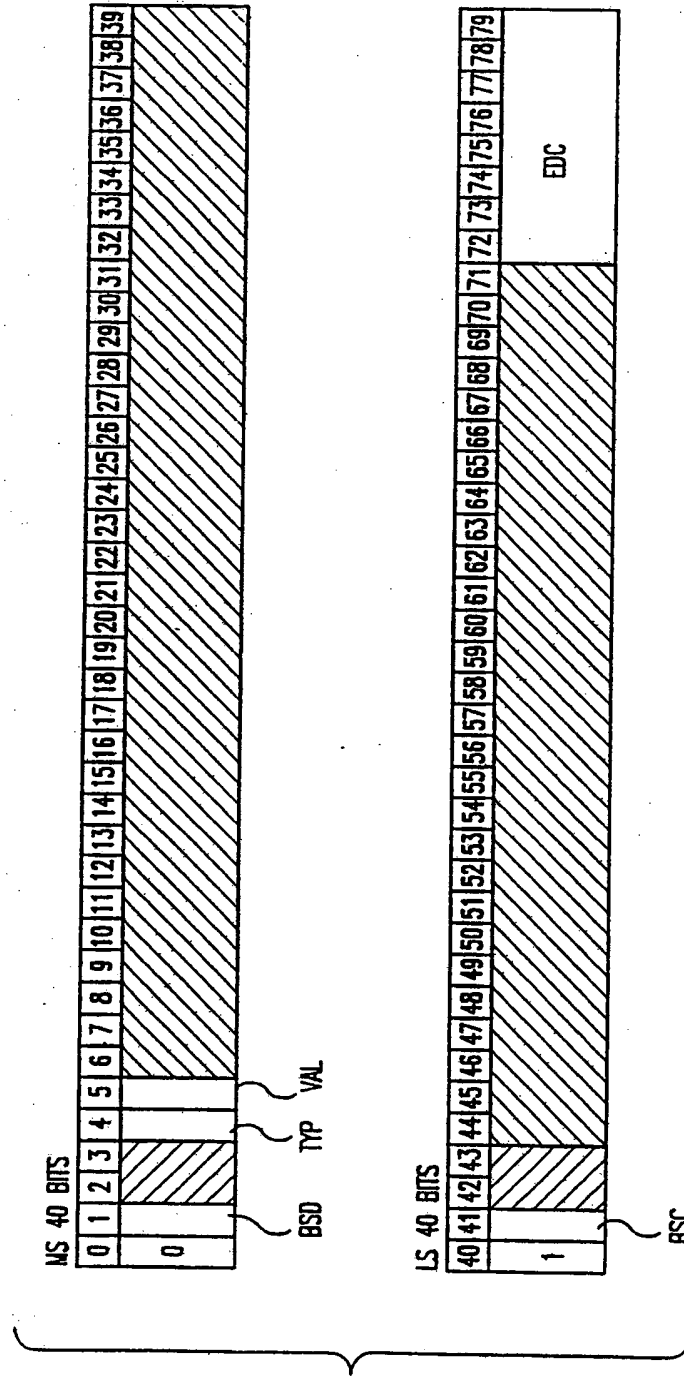
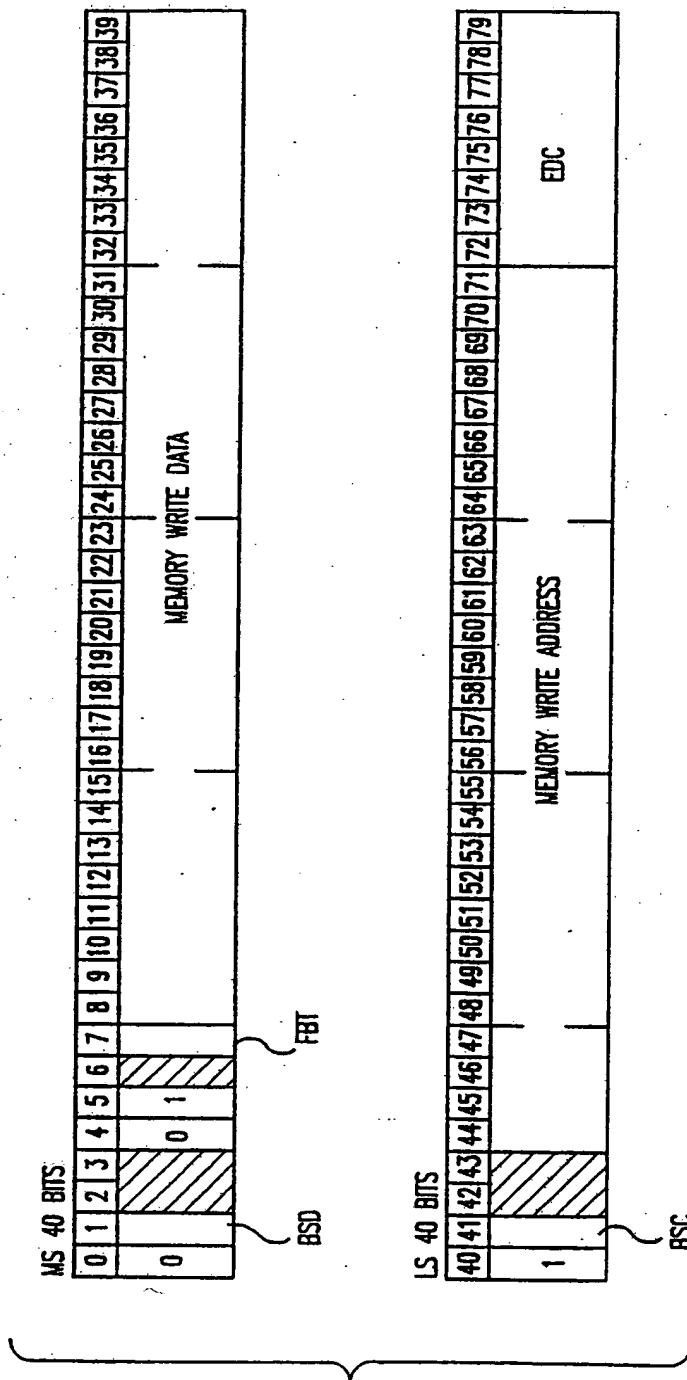


FIG.12



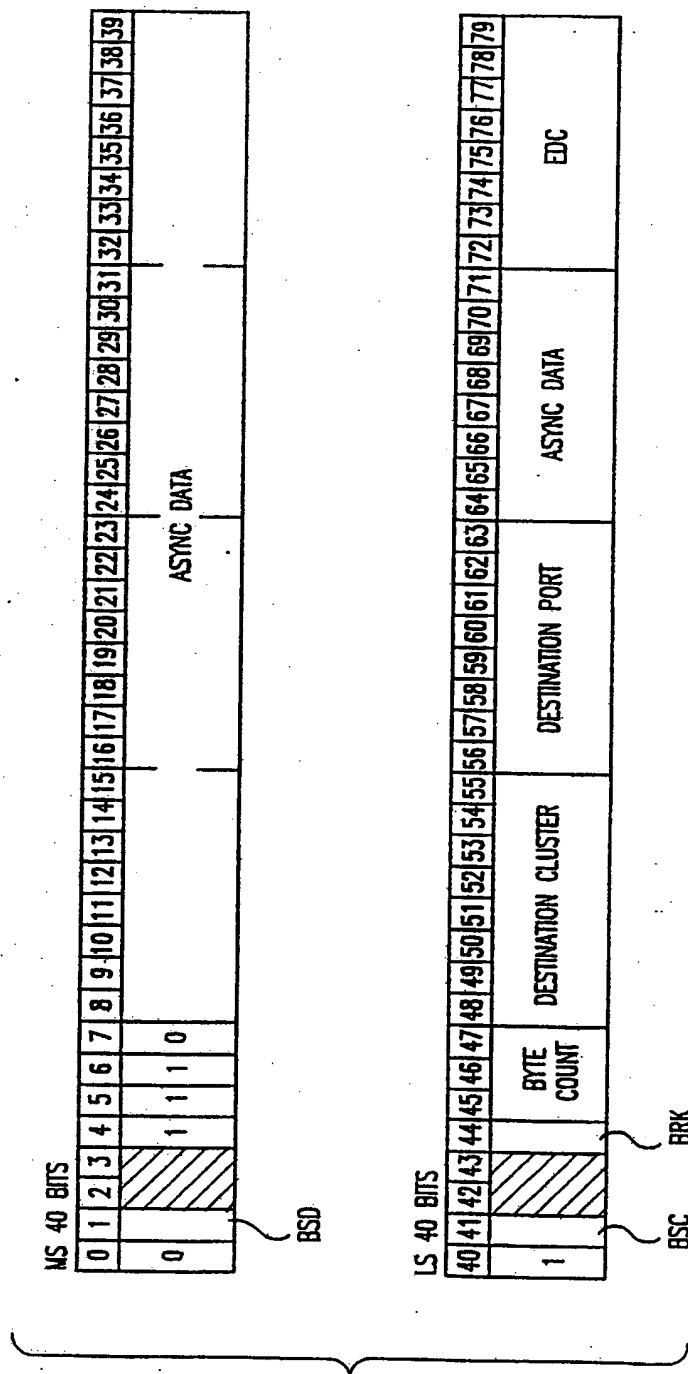
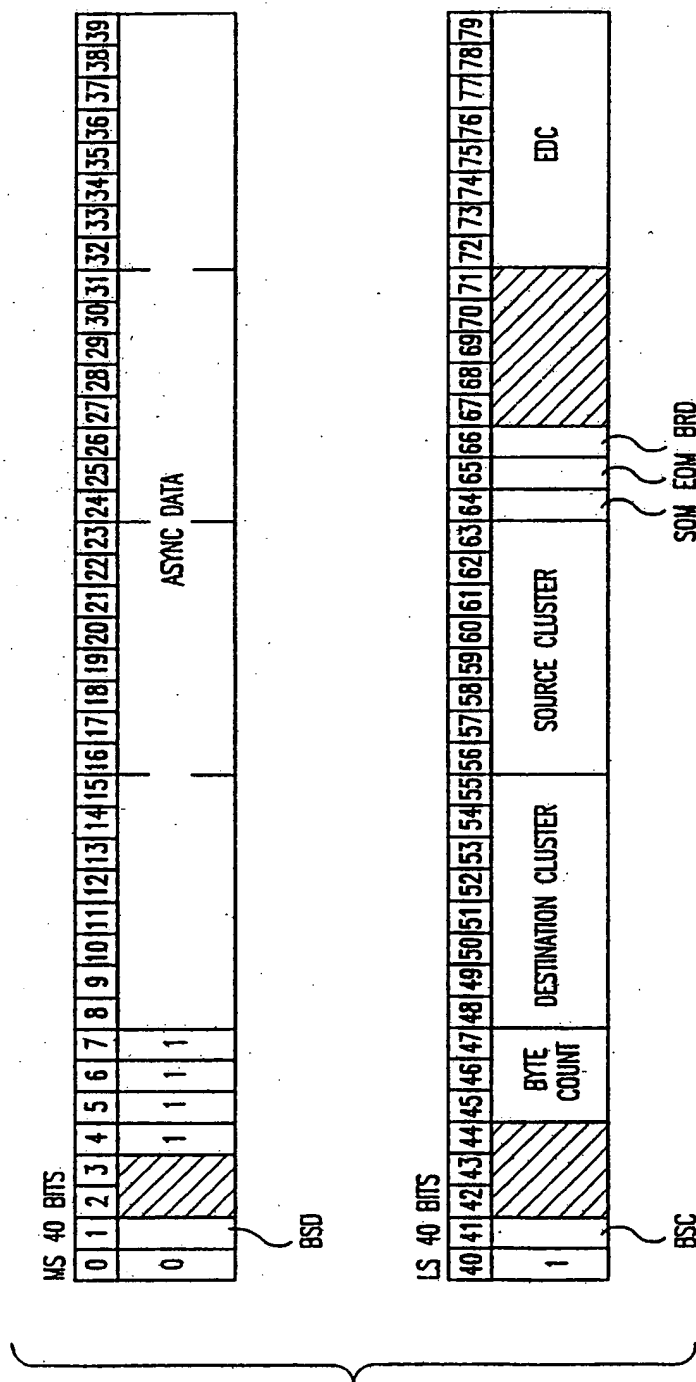


FIG.14



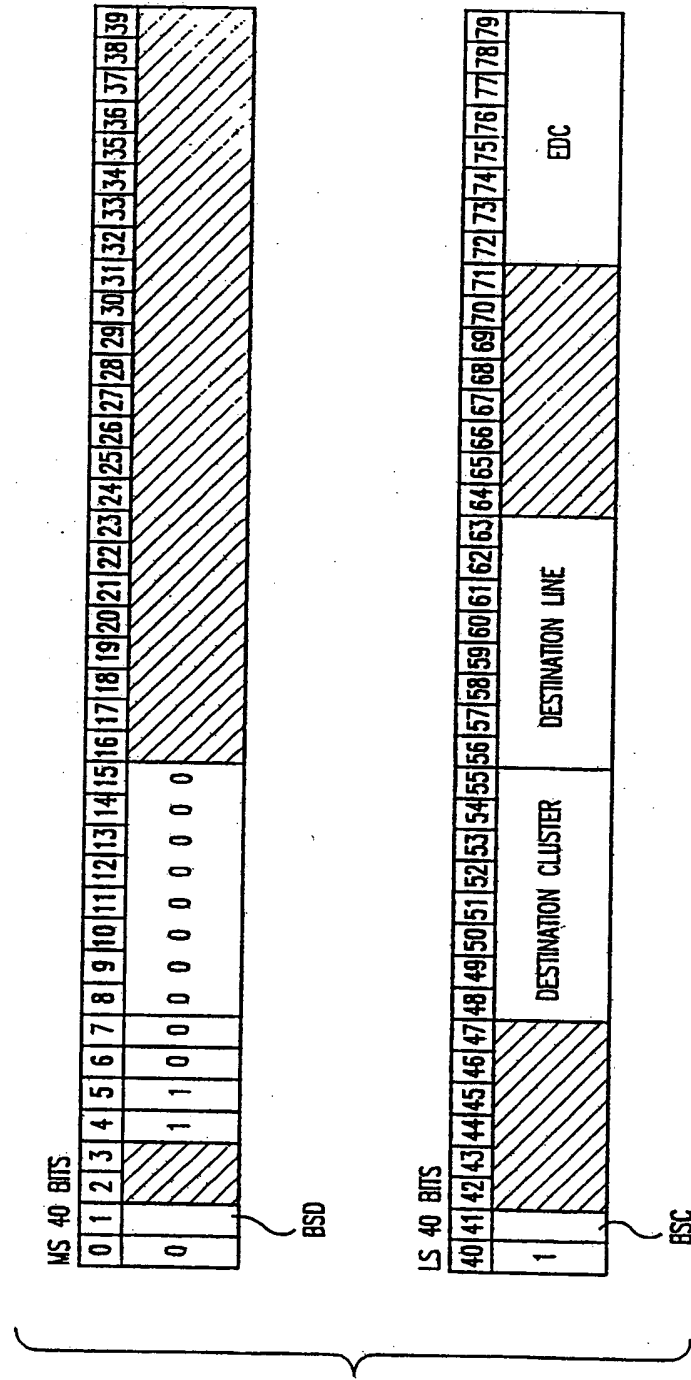


FIG.16

17/31

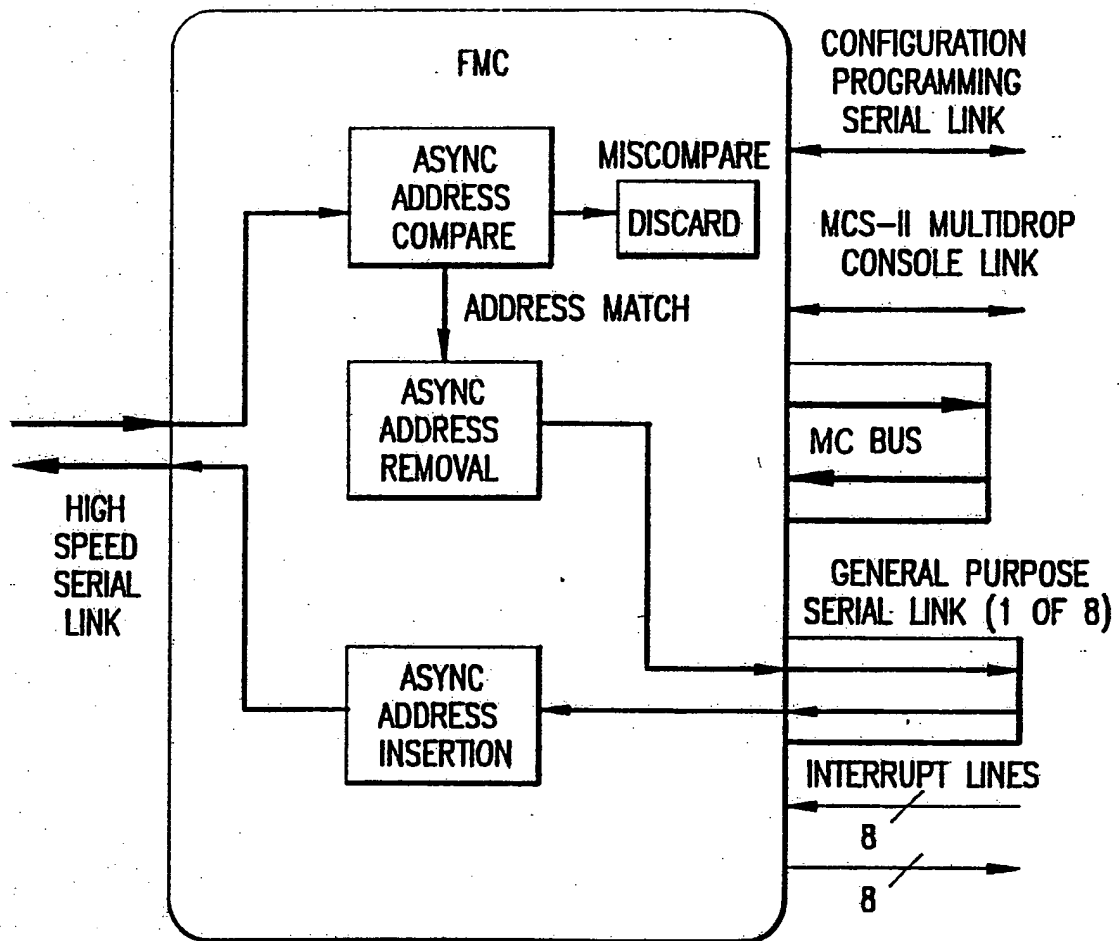


FIG.17

18/31

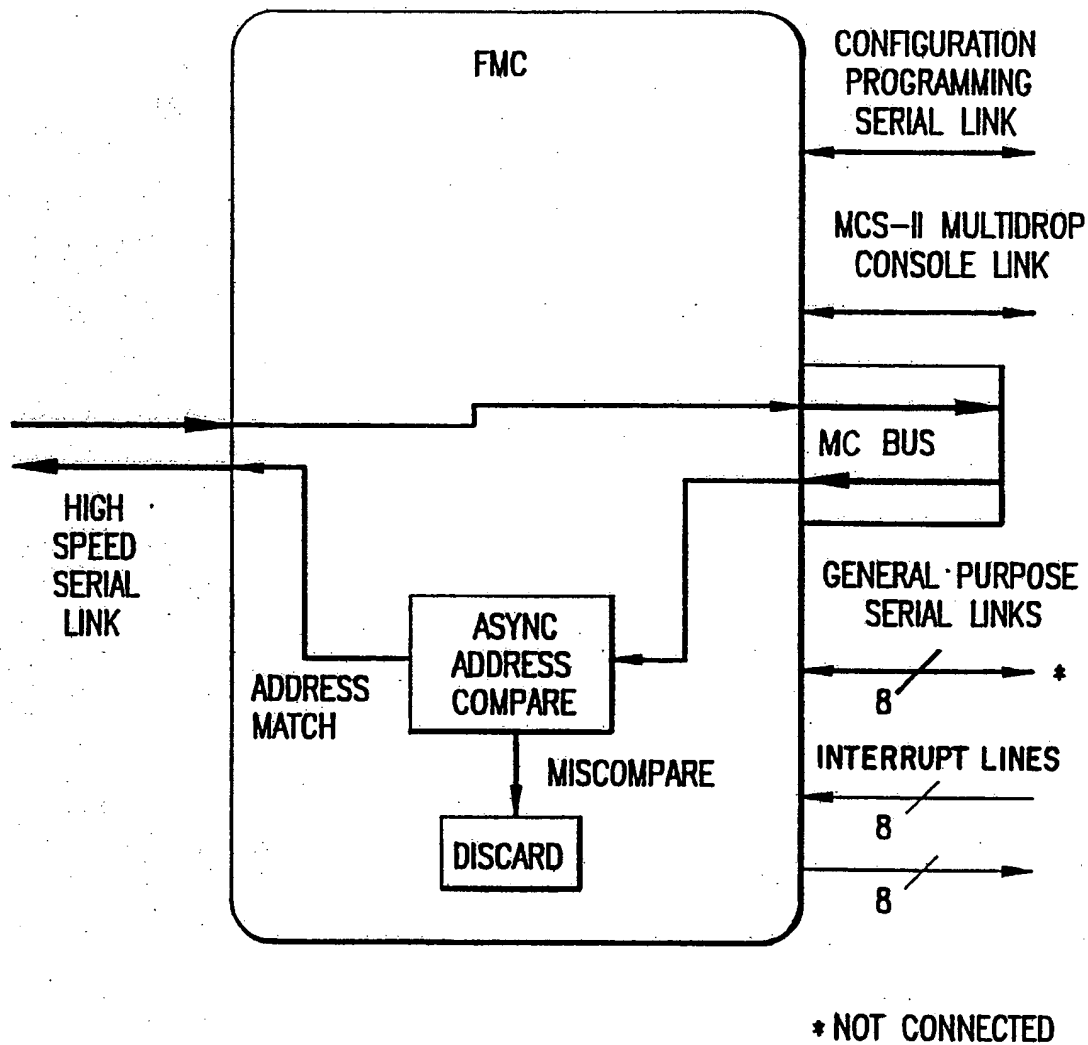


FIG.18

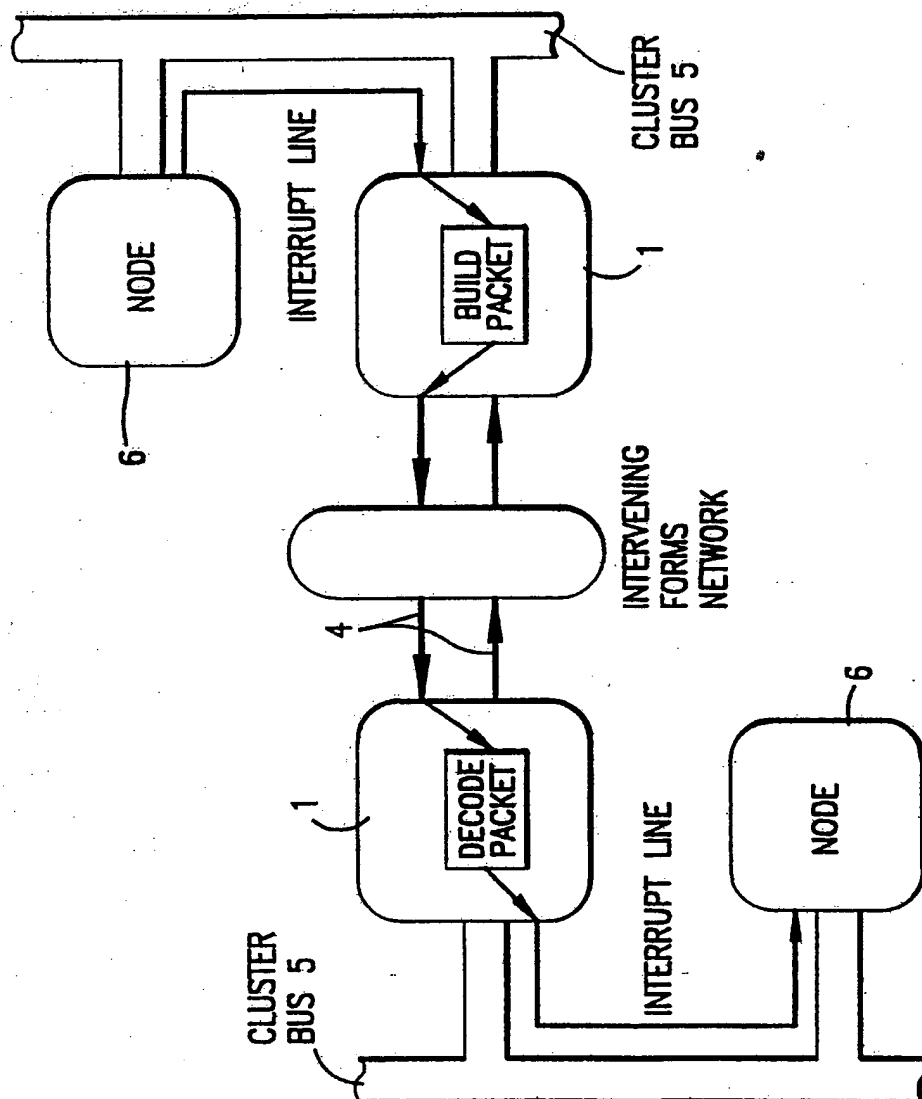


FIG.19

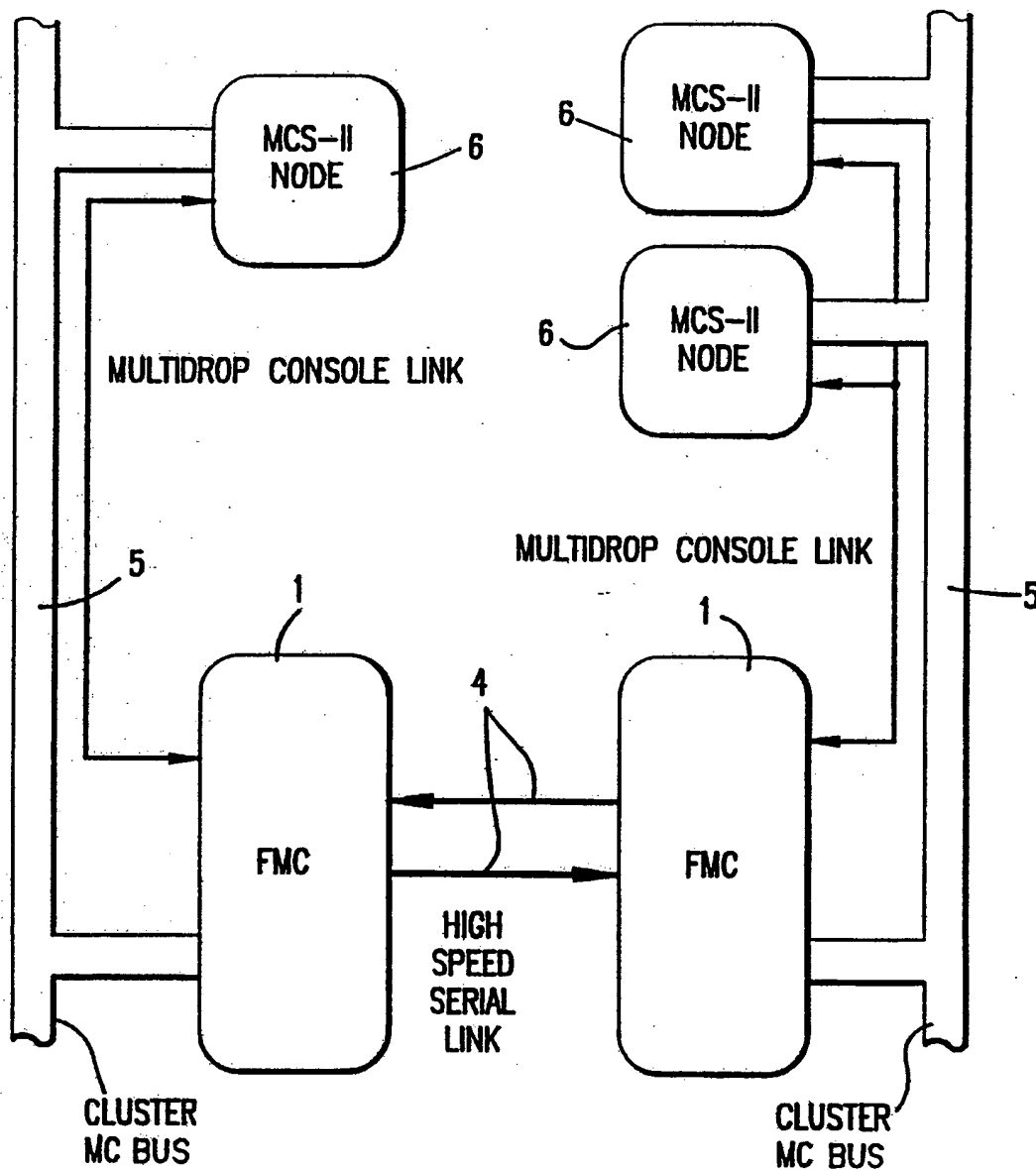


FIG.20

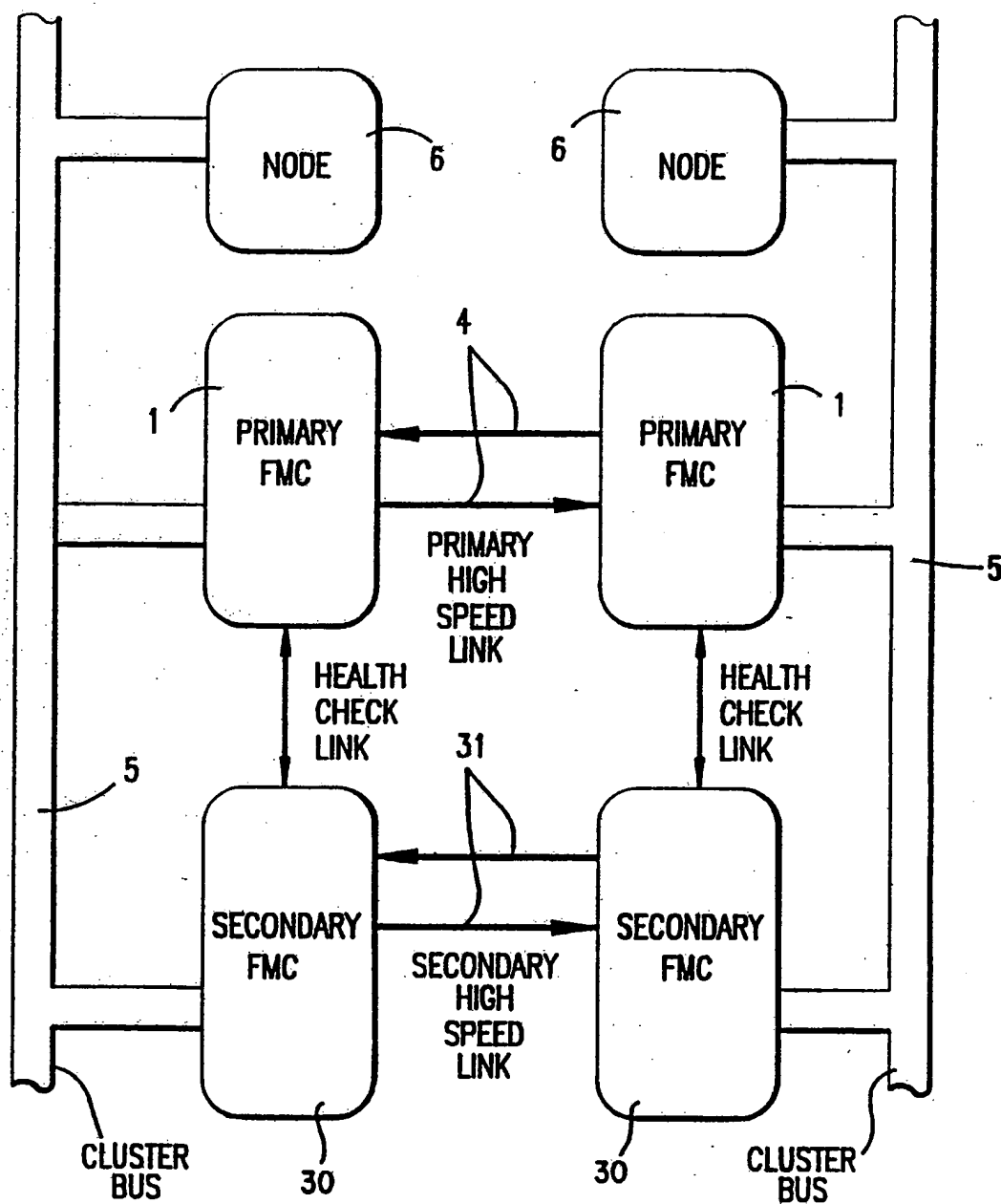


FIG. 21

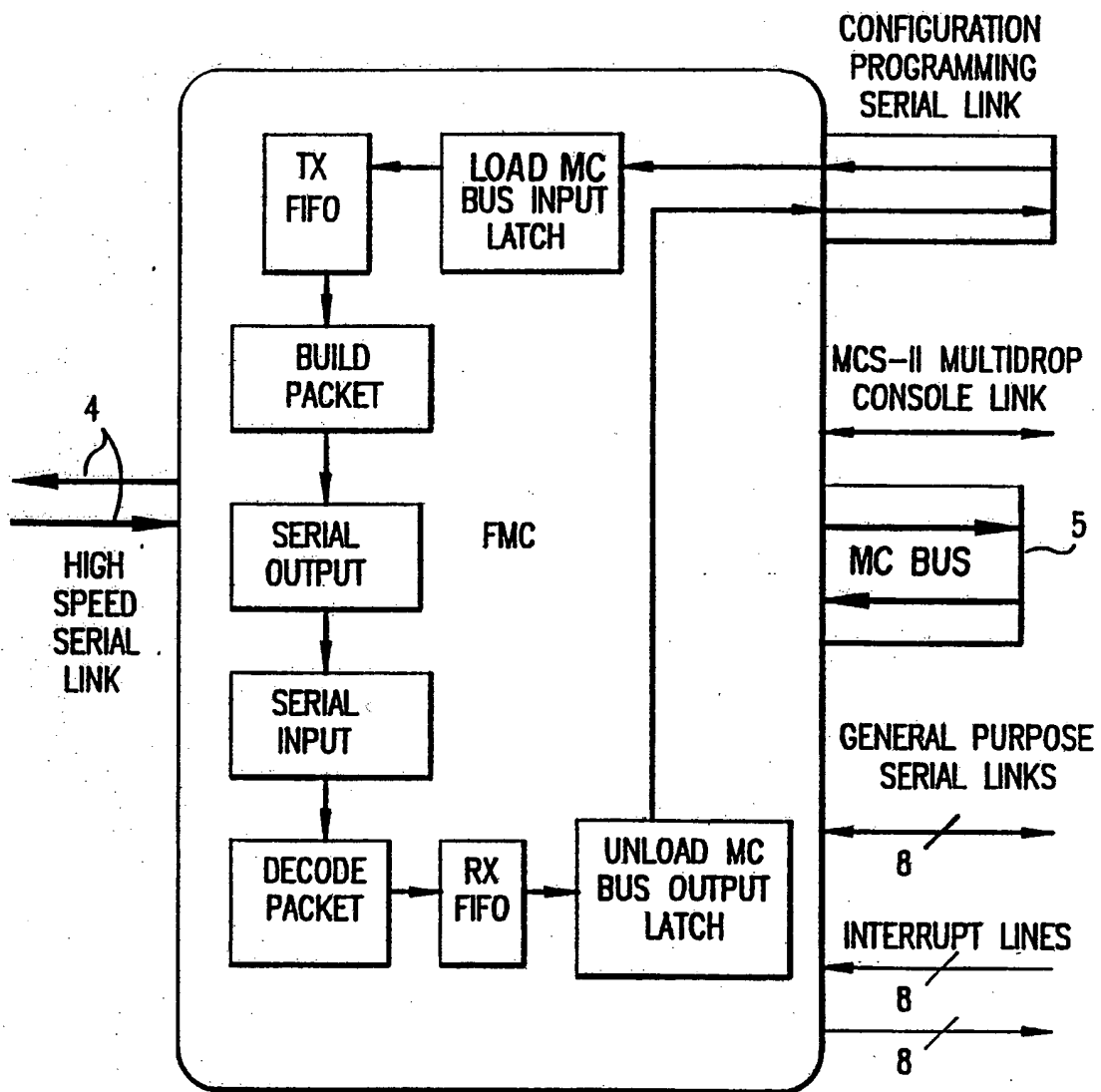


FIG.22

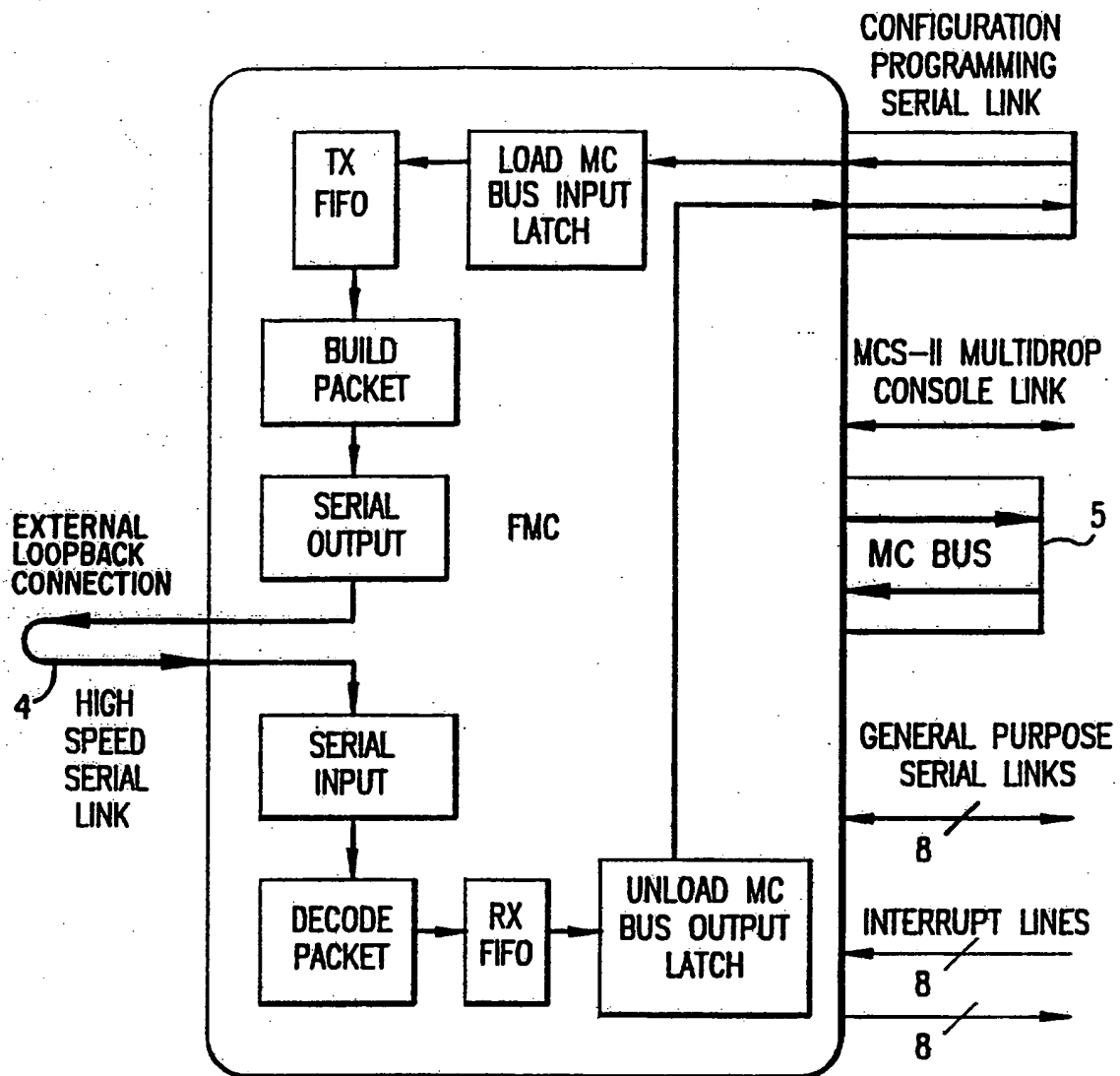


FIG.23

24/31

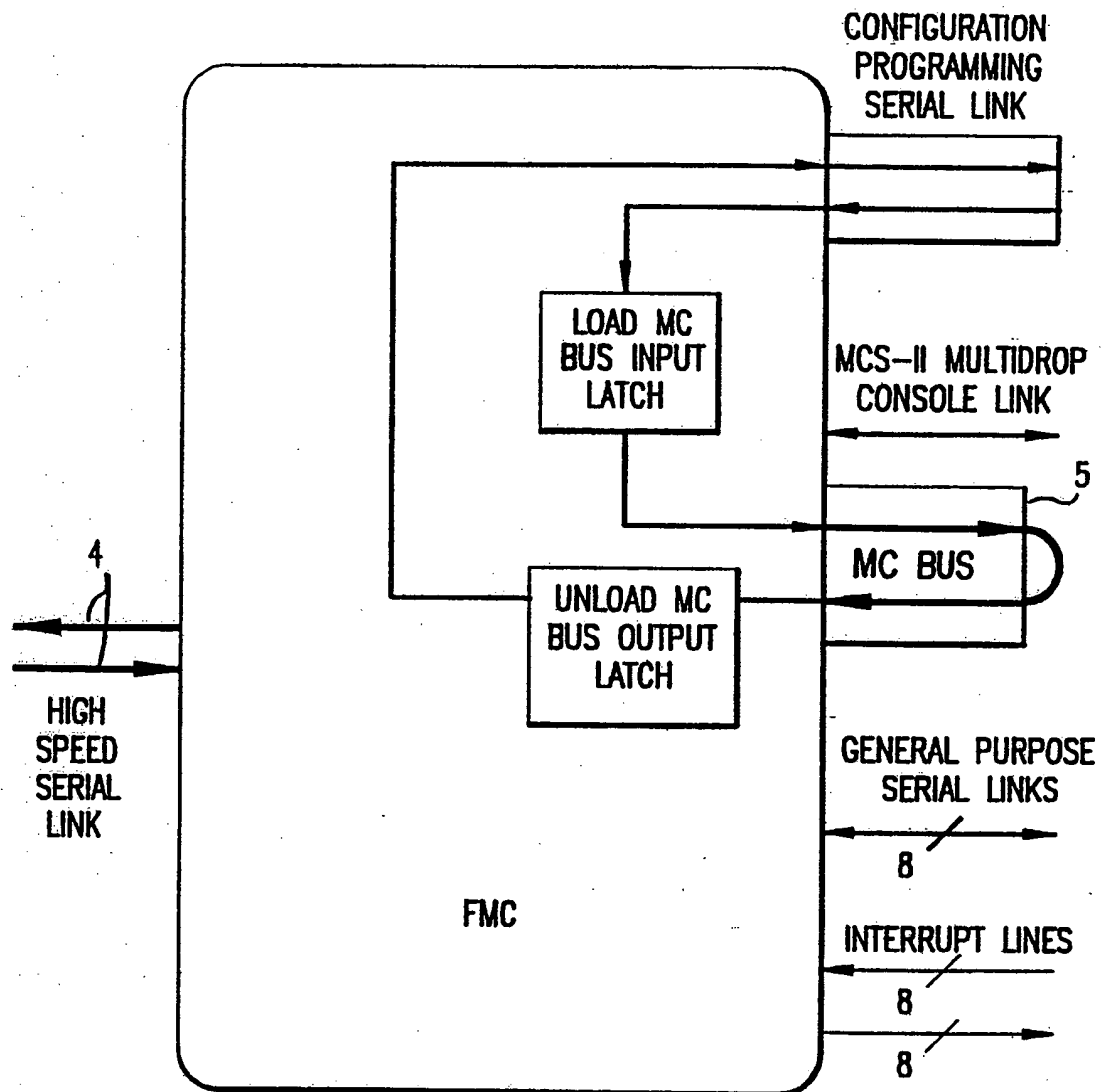


FIG. 24

25/31

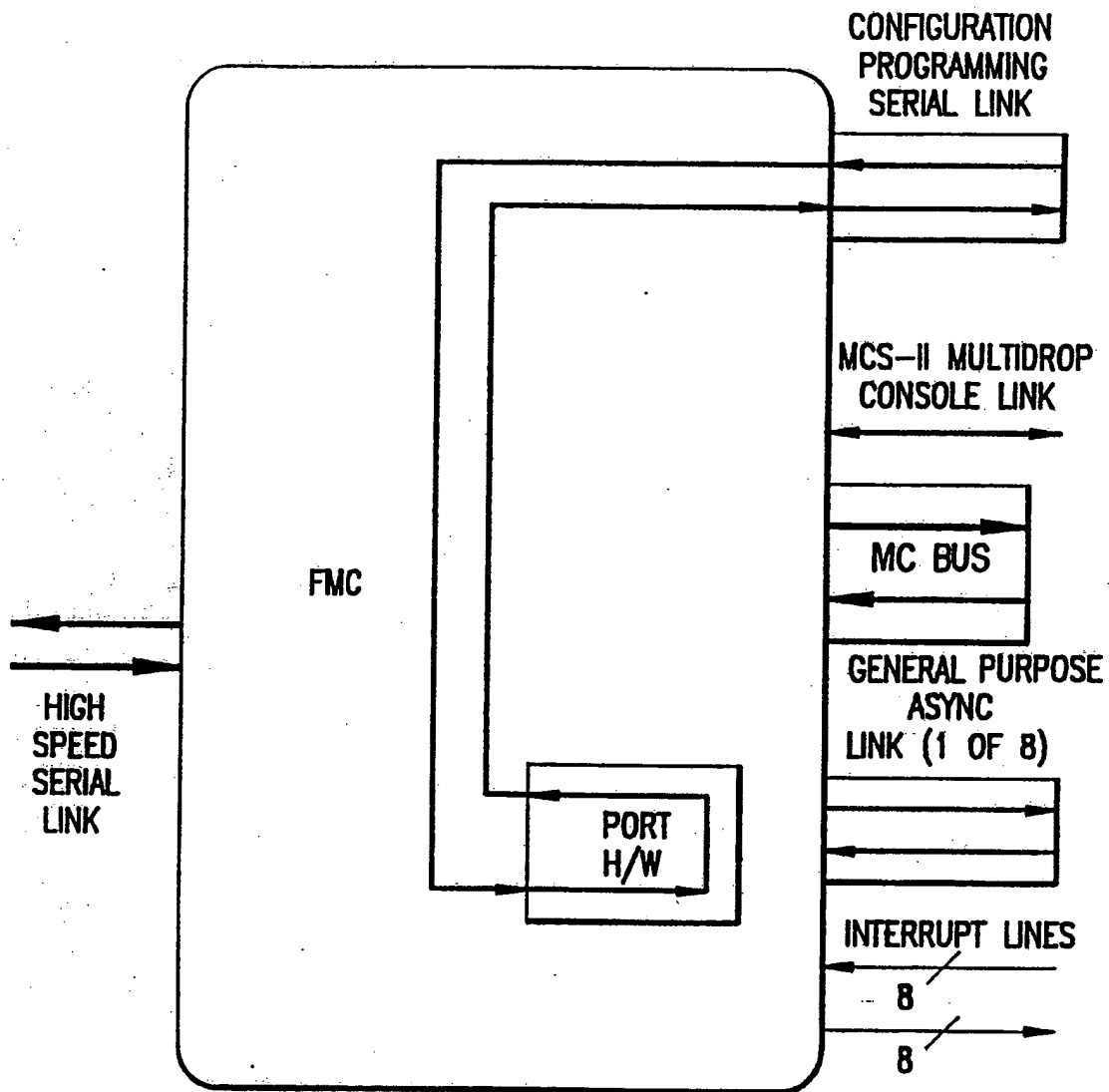


FIG. 25a

26/31

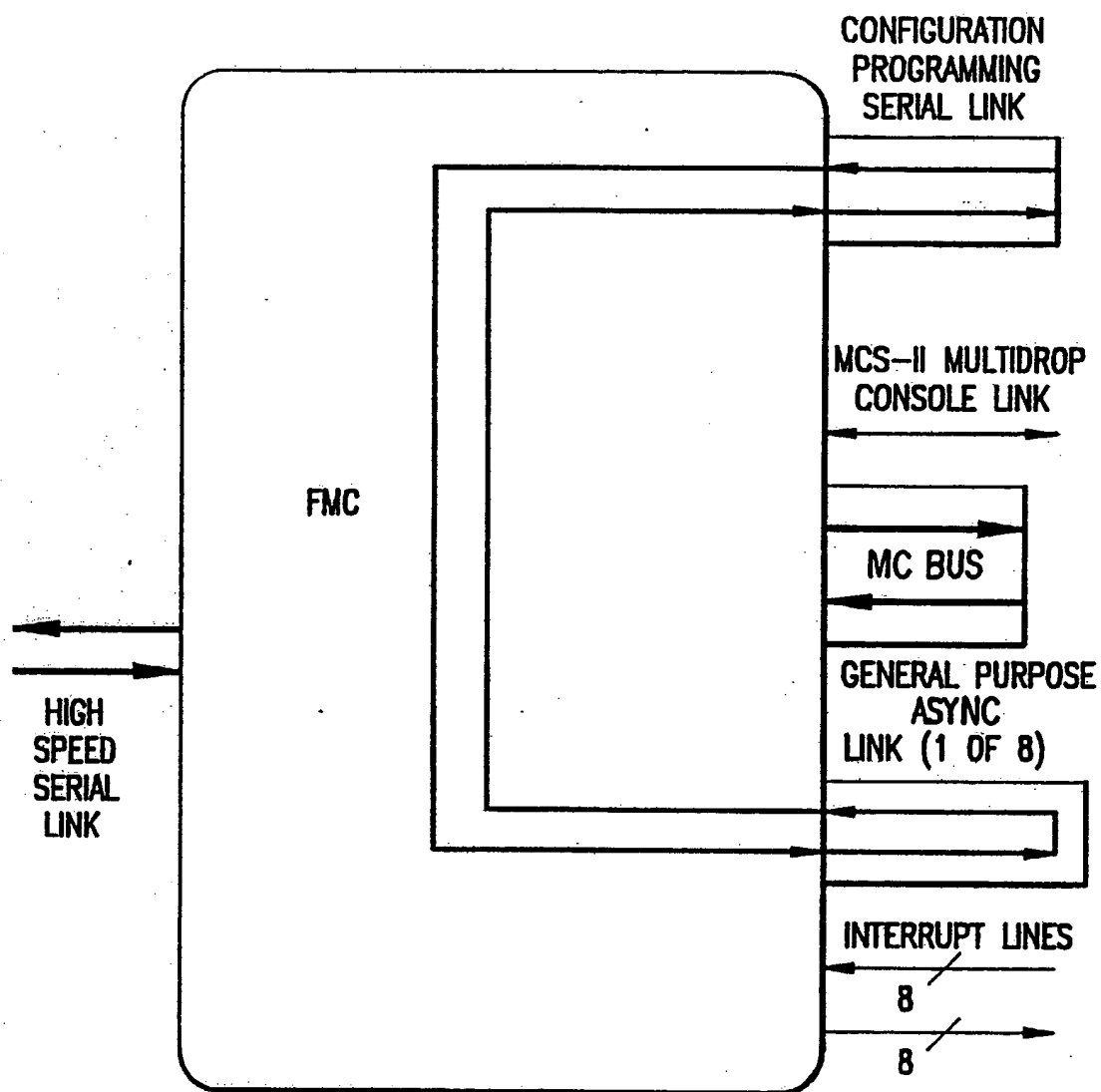


FIG.25b

27/31

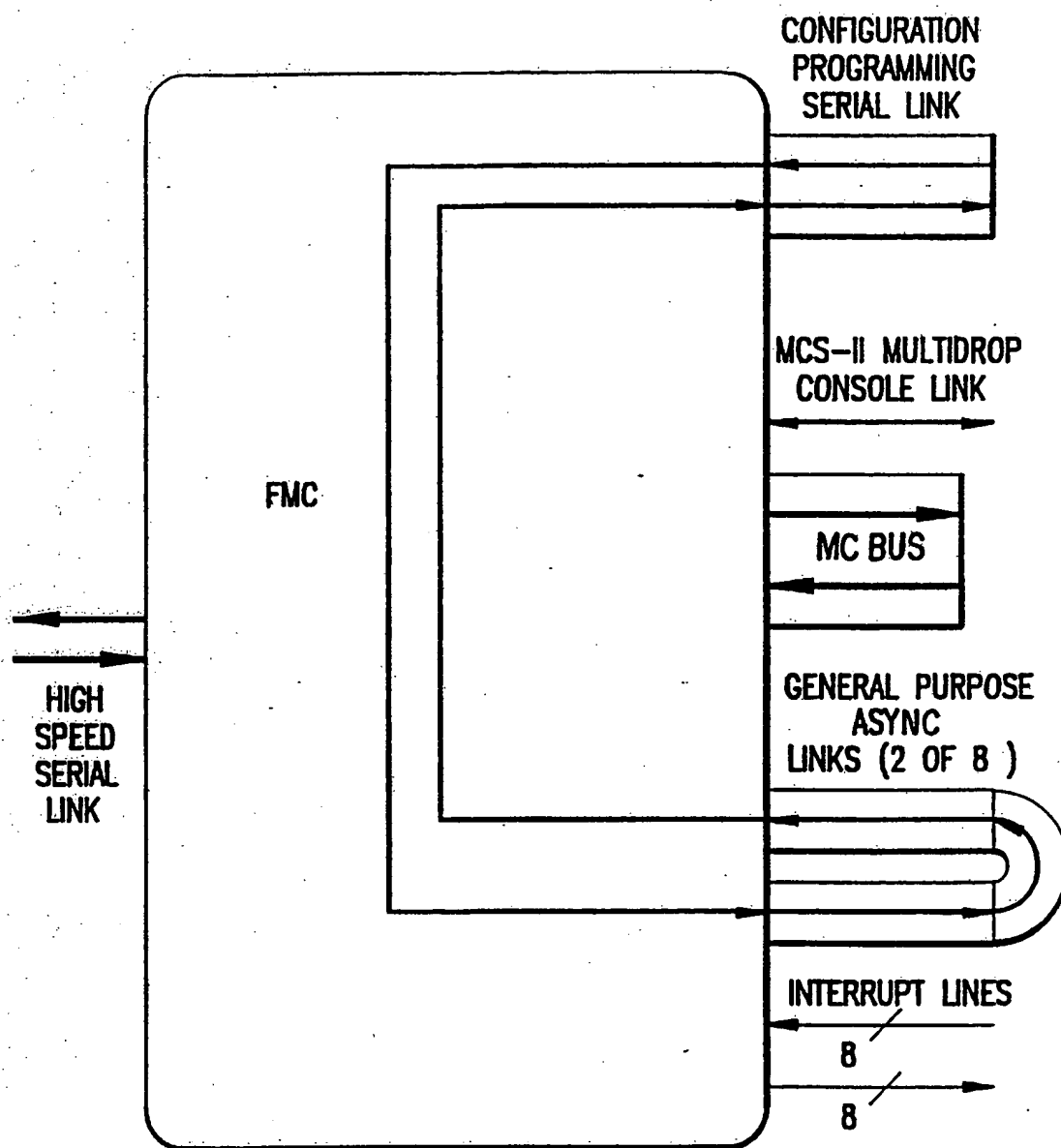


FIG.25c

28/31

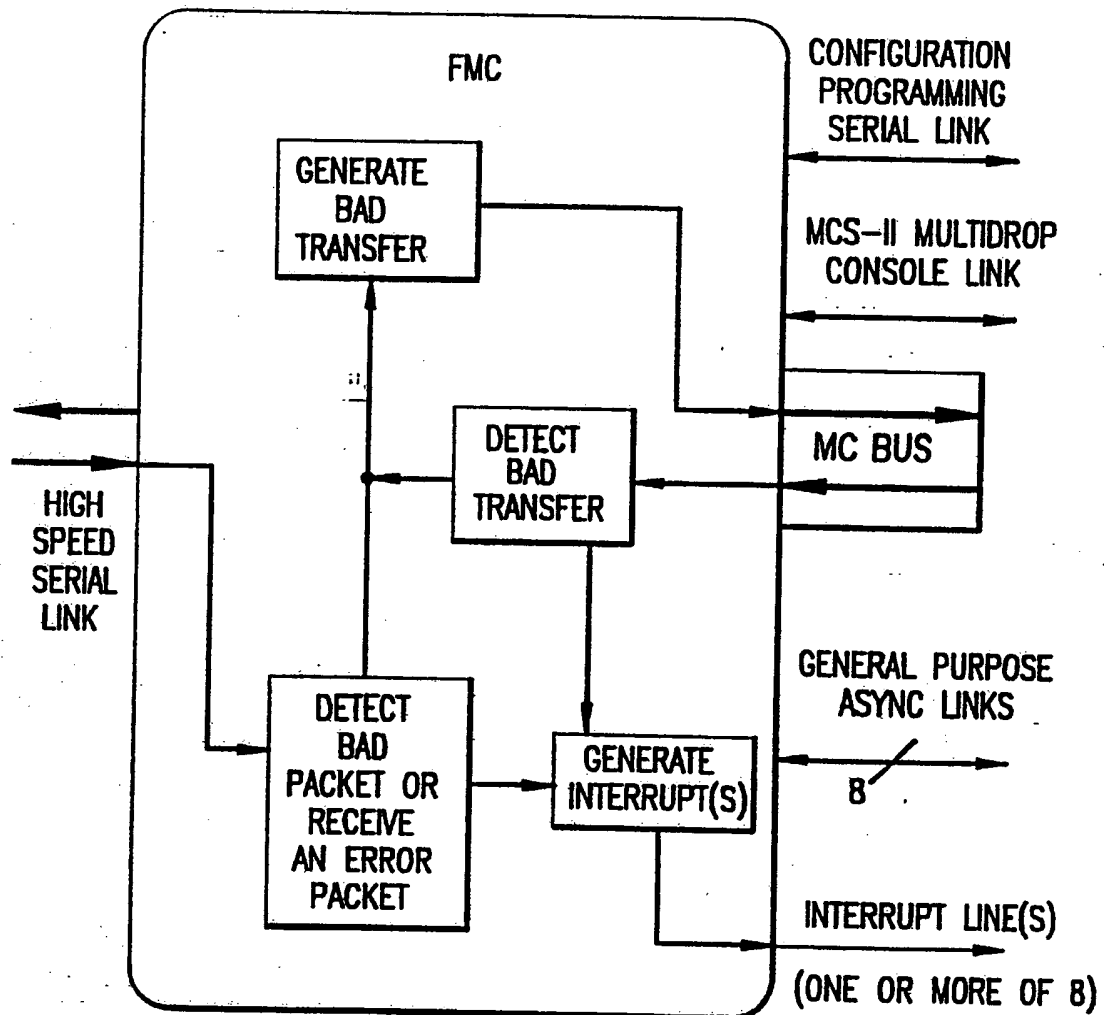


FIG.26

29/31

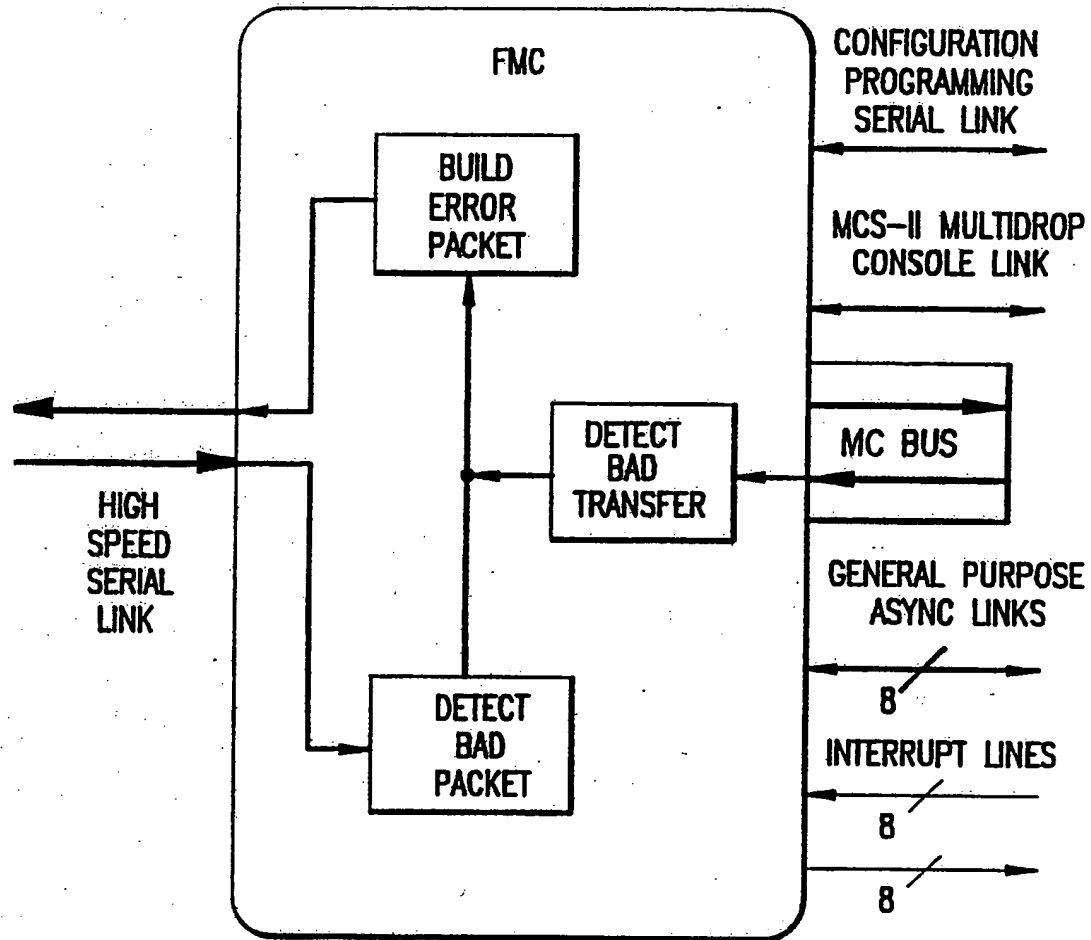


FIG.27

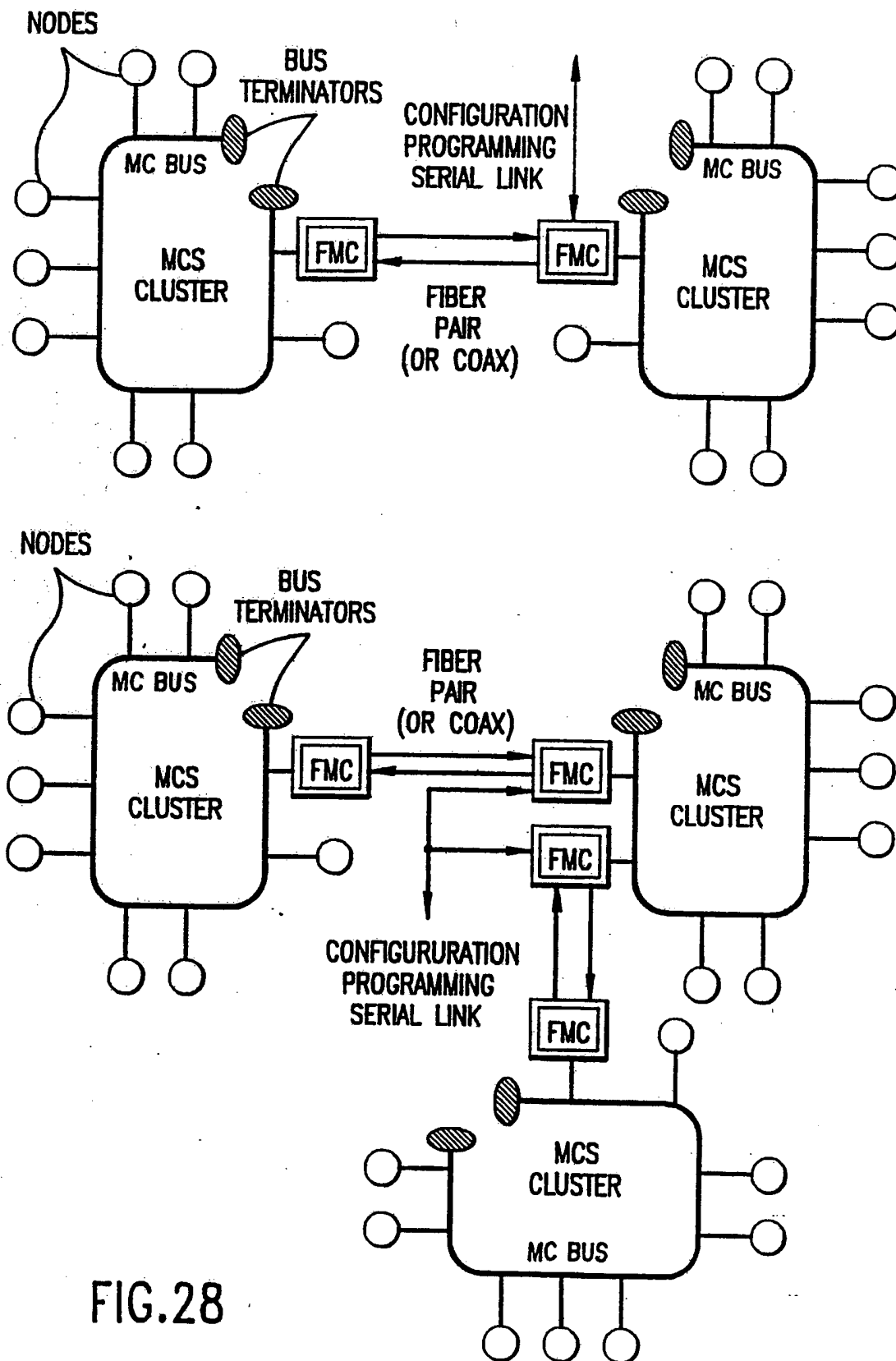


FIG.28

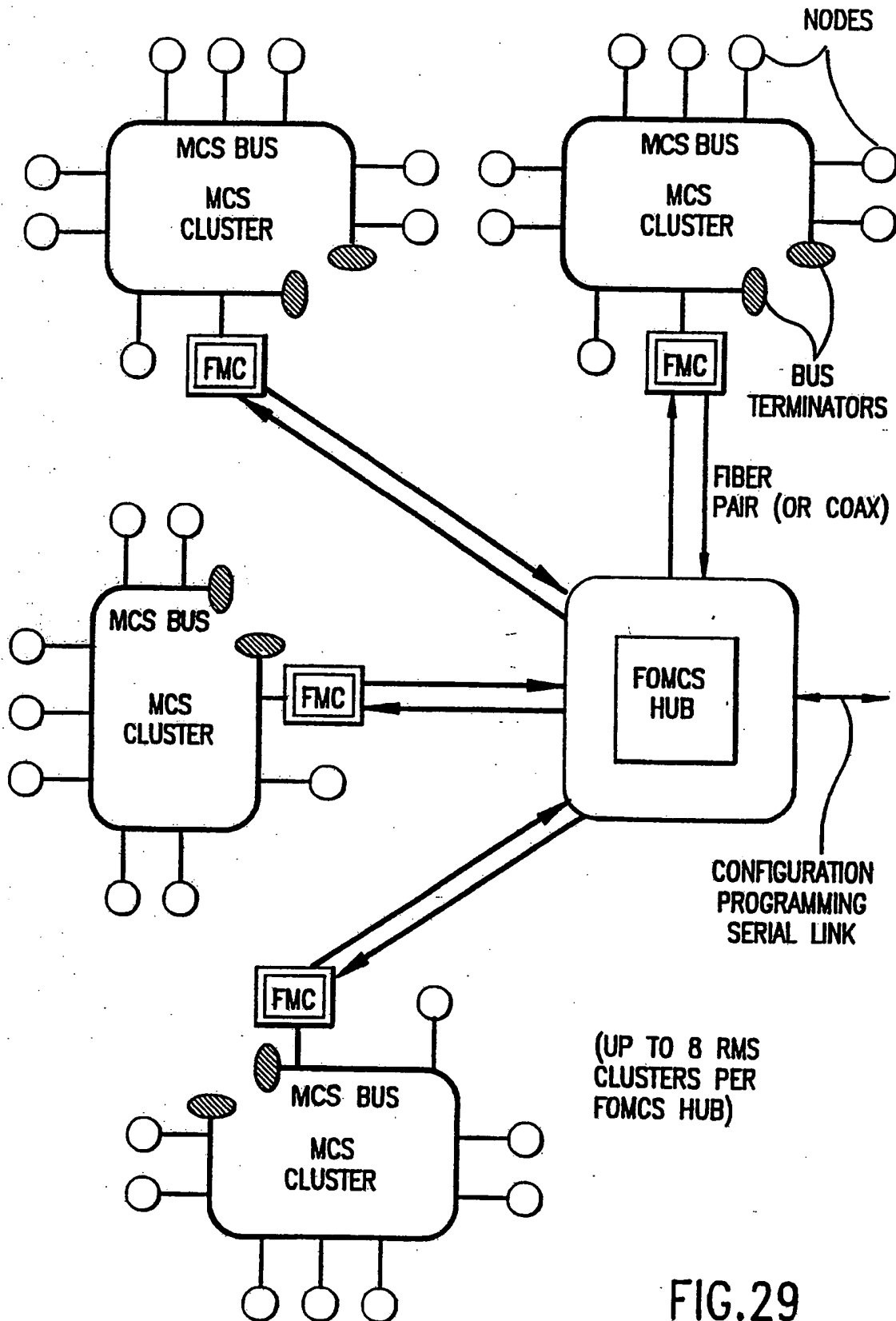


FIG.29

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US93/02839

A. CLASSIFICATION OF SUBJECT MATTER

IPC(5) : G06F 13/00

US CL : 395/200

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 395/325, 395/575, 395/500

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
Please See Extra Sheet.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|-----------|---|-----------------------|
| Y,P | IEEE Network, May 1992, Vetter et al., "Network Supercomputing", p.38-44. | 1-4,8,5-7 |
| Y | IEEE, 1992, McFarland et al., "HP's Link Interface Chipset Lu Serial HIPPI", p.229-233 | 1-4,8,5-7 |
| Y | US,A, 4,466,063 (Segawa et al) 14 August 1984 See the entire document | 1-10 |
| Y | US,A, 4,430,699 (Segawa et al) 02 February 1984 See the abstract | 1-10 |

☒ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

| | | |
|--|-----|--|
| * Special categories of cited documents: | * T | later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention |
| * A* document defining the general state of the art which is not considered to be part of particular relevance | * X | document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone |
| * E* earlier document published on or after the international filing date | * Y | document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art |
| * L* document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) | * Z | document member of the same patent family |
| * O* document referring to an oral disclosure, use, exhibition or other means | | |
| * P* document published prior to the international filing date but later than the priority date claimed | | |

Date of the actual completion of the international search

11 MAY 1993

Date of mailing of the international search report

07 JUL 1993

Name and mailing address of the ISA/US
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231

Authorized officer

D. DINH

Facsimile No. NOT APPLICABLE

Telephone No. (703) 305-9600

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US93/02839

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|-----------|--|-----------------------|
| Y | US,A, 4,269,221 (Lippmann et al) 06 September 1988 See Fig. 1 | 1-10 |
| A | US,A, 4,935,894 (Ternes et al) 19 June 1990 See the entire document | 1-10 |
| A | US,A, 4,342,083 (Freedman et al) 27 July 1982 See the entire document | 1-10 |
| A | US,A, 4,396,995 (GRAM) 02 AUGUST 1983 See the entire document | 1-10 |

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US93/02839

B. FIELDS SEARCHED

Electronic data bases consulted (Name of data base and where practicable terms used):

(FILE 'USPAT' ENTERED AT 11:01)

L1 7 S 4396995/UREP
L2 510 S 395/200/CCLS
L3 24433 S MEMORY (P) WRITE#
L4 175 S L3 AND L2
L5 192573 S OPTIC?
L6 25 S L5 AND L4

Pro Quert (IEEE CD ROM) - "HIPPI"